電腦時代

COMPUTING AGE

9

微型電腦用家雜誌

HK$8.00




TAXAN RGB「幻彩系列」
彩色電腦螢幕
色彩艷麗完美無瑕
● 能顯示任何顏色組合
率先採用「線性影像放大系統」
● 絕不干擾磁碟及其他電器
設計符合多項美國嚴格標準
● 適用於各款電腦
備有多種附件，適用於下列電腦：
RGB（紅綠藍）彩色介面
APPLE II¹ APPLE IIe¹
介面接駁線
APPLE III¹ IBM PC² BBC³ NEC PC⁴
FUJITSU FM-7⁵ COMMODORE
歡迎訂造特別介面接駁線
1. APPLE IS A TRADE MARK OF APPLE COMPUTER INC
2. IBM IS A TRADE MARK OF INTERNATIONAL BUSINESS MACHINE CORP
3. BBC IS A TRADE MARK OF ACORN INTERNATIONAL LTD.
4. NEC PC IS A TRADE MARK OF NEC CORP.
5. FUJITSU FM-7 IS A TRADE MARK OF FUJITSU LTD.
6. COMMODORE IS A TRADE MARK OF COMMODORE ELECTRONICS LTD.
RGB vision-III
TAXAN
享有代理商保證
才是專業性保用
總代理
VINCENT COMPUTER CENTRE LTD.
雲生電腦有限公司
九龍尖沙咀柯士甸路22-26號好兆年行608室

PROMAS 305 3吋超小型磁碟機

小巧強勁



- 143K儲存量，完全適用於任何APPLE* 操作系統(OPERATING SYSTEM)，並可與傳統5吋磁碟機及DISK DRIVE CARD共用。
- 體積爲普通5$\frac{1}{4}$吋磁碟機之三份一，易於放置。
- 全部日本製造及裝嵌。
  @HK$1350

使用3吋盒式磁碟之優點：

- 特有「保護閘門」(SHUTTER MECHA-NISM)，可防塵及手指觸摸，並有機械式WRITE PROTECT開關裝置，更爲方便。
- 絕密封套，令磁碟耐用十倍及眞正可作兩面使用，合符經濟原則。
- 每張磁碟均有獨立硬膠盒盛載，加强保護磁碟及方便整理。
  每盒十張只售HK$280

價格如有變動，恕不另行通告。

PROMAS ©

*Trademark of Apple Computer Inc.

總代理：
VINCENT COMPUTER CENTRE LTD.
雲生電腦有限公司
九龍尖沙咀柯士甸路22－26號好兆年行6字樓 電話：3-7211288-0 電訊：HX 38316 VCC HX

LEE'S ELECTRONIC CO.

李氏電子公司



美國IRISH
首次推出
處女$5\frac{1}{4}$"磁碟
本公司獨家經銷
歡迎批發·零沽

SUPER SOFTSWITCH II

* Work on all 80 column Graphic Programmes, e.g. GBASIC, PASCAL, DGRAPH
* Work on pfs. DOLLARS & SENSE, all CP/M programmes

SPECIAL OFFER!

SPEECH CARD II

中文，英文說話卡，更可幫你讀出 CATALOG，DOS COMMAND，BASIC PROGRAM，外接式高品質小型喇叭箱，聲音響亮清晰Software＋Manual

NEW JOY-PORT III

特別設計全新綫路，絕對可玩全部 Game，可以選擇AUTO FIRE，配合QUICK SHOT JOYSTICK，

注意

QUICK SHOT JOYSTICK
如果唔加JOYPORT CARD
唔可以玩LADY TUT，或者
將來新Game

COMPUTER DATA RECORDER FOR APPLE COMPUTER

電腦專用錄音機獨有SAVE和LOAD鑑聽和指示 AUTO STOP帶長 COUNTER保用三個月

*"香港區總代理，歡迎批發"

私人電腦的新概念：記憶咭

可以說：這張記憶咭將會廣泛受到歡迎

沒有鍵盤及毋需電源供應器的神奇微型電腦，未來每個人口袋裡都會攜帶一個。這種記憶咭（又稱爲醒目咭或晶片咭）形狀和體積與信用咭一樣，內藏有微型電腦。目前美國及歐洲正進行市場試驗，記憶咭用來作紀錄及顯示資訊，並力求做到資料保密完美無暇。

另一種記憶咭並沒有內置微型電腦，而有一層薄膜，使資訊變成密碼紀錄下來。還有一種並不是咭的形狀，就像一條塑膠鎖匙，由有非揮發性的半導體記憶容量。（見照片1）。

至目前爲止，記憶咭基本應用在於電子資金交流（ELECTRONIC FUNDS TRANSFER〔EFT〕）範疇上。法國已嘗試幾個程式，以利用記憶咭作爲"電子貨幣"，例如，電視傳真機的用家可以用記憶咭安坐家中購買物品及服務。在意大利，一家半導體公司正生產支付電話費的原模記憶咭，你可以購買儲值的記憶咭，每打一次電話，費用就會在記憶咭中扣除，直至用完全部價值，記憶咭就作廢。

記憶咭科技

記憶咭是使用三類科技：單晶片記憶咭、多晶片記憶咭及光條記憶咭。單晶片記憶咭有一個完整電路，包括單晶片電腦有一個 PROM 最高儲存四千個位元資訊。PROM 是用作紀錄及操作記憶咭的"交易"。這種咭，是隨時準備扔棄的，因爲記憶量注滿之後，便不再紀錄新的交易。爲了補救這個缺點，生產商正研究利用電子塗抹可編程序記憶（EEROM）。不過，此科技雖然可以消除了記憶量"爆滿"的難題，但另方面却可以增加了保密的可靠性及資料完整的問題。

記憶咭用超過一個 IC的稱爲多晶片記憶咭。它比較先進，因爲可儲存大約一萬六千個位元資訊。經包裝技術改良及 IC科技的發展，有較大記憶容量的多晶片咭在應用上顯然更勝一籌。

光條記憶咭有獨特的表層，有如錄影碟上的薄層，光條的面積如果是信用咭的磁條的一樣大小，可儲存一千六百萬位元資訊。而磁條只能儲存一千七百個位元資訊。表1是不同記憶咭儲存密度的比較表。

記憶咭能夠與微型電腦配合使用的稱爲主動記憶咭，這種咭有一定程度的內在智能，使它與外間世界互爲作用時，可以作出決定。使資訊保密或在交易過程中，每個程序都恰當無誤。

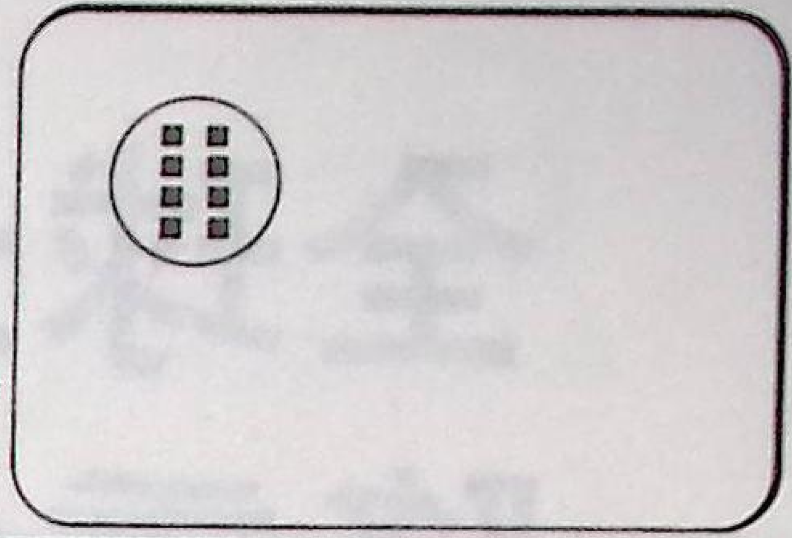
另方面，被動記憶咭（例如光條記憶咭）並沒有主動智能，互爲作用完全依靠外間世界。不過被動記憶咭並非完全沒有保障，因爲它的大記憶容量，令它可以儲存分辨擁有者的資料。

記憶咭閱讀器或交易終端機的設計



照片1　四種不同記憶咭，下三種爲主動（ACTIVE）記憶咭，內置微型電腦，最上一張被動（PASSIVE）記憶咭

資料儲存媒介——記憶咭



圖（1）一張主動記憶咭接觸點典型的方排。最少需要6至8個接觸，因爲資料是串行傳送的。

	比特	（BYTES）	翻用記憶
光條	16M	2M	否
PROM	128K	16K	否
EEPROM	64K	8K	能夠
EPROM	128K	16K	否
磁條	1.7K	212	能夠

表（1）記憶咭儲存量比較表

端視乎記憶咭是被動或主動。終端機只需I／O（輸入／輸出）裝置及電源供應器，因爲記憶咭已內置有微型電腦（見照片2）

光條咭則需要較爲複雜的鐳射資料交換系統去了解光條咭上的孔型密碼。

記憶咭結構

主動記憶咭的結構一般外型大小如信用咭，內置一個微型電腦。微型電腦透過咭表面的接觸點與外間世界連繫起來，接觸點只需幾個，因爲只有能源及資料傳送於記憶咭及交易終端機之間，圖1顯示單晶片記憶咭的外型，圖2則顯示單晶片記憶咭的一般結構。

多晶片記憶咭的結構與單晶片記憶咭差不多，除了有較大記憶容量及每一個IC可以是個別次系統SUBSYSTEM，顯示多晶片記憶咭的記憶及結構。

資料是經由I／O閘傳送，這個通度相似於收發器，而由"看門狗"（保安官）監察。它的作用是弄清楚輸入／輸出的交易是否有效。例如，連次交易都使用不適當的密碼，"看門狗"察覺後，便會永久性地關上輸出／輸別大閘，並宣佈該咭失效，不能再作任何交易。

單晶片及多晶片的記憶部份是四種記憶的組合：現成ROM（READ—ONLY MEMORY唯讀記憶），RAM（隨機搜索閱／讀記憶）作臨時資料儲存，EEROM或PROM（儲存交易紀錄）。

光條記憶咭的結構則較爲簡單，咭上加了特別的物質。驟眼看來，普通人難以分辨出磁條信用咭及光條記憶咭。光條實際上有四層：保護膜、反射層、不反射層及隱蔽層（見圖4），利用鐳射光在反光層上燒成無數的微孔，資料便化爲密碼。再利用鐳射掃描光條，便會產生出連串的元數資料。圖5顯示鐳射掃描光條時的情況。

記憶咭終端機

交易終端機是通訊站，用來與記憶咭交換資訊。基本上，它包括兩部份：記憶咭介面及接上鍵盤的顯示器，記憶咭介面有兩種：一爲光條記憶咭，另一爲單晶片及多晶片記憶咭。

光條記憶咭介面藏有電腦控制鐳射資料交換掃描器，掃描器可以翻釋記憶咭上的微孔成爲有意思的資料，由於此種介面，要有電腦及鐳射掃描器，故此生產成本比單晶片及多晶片記憶咭的介面昂貴得多。

單晶片及多晶片記憶咭終端機介面只需是機械式的裝置使記憶咭與終端機接觸點連結起來，供應記憶咭能源及與鍵盤及顯示器作連串的通訊交往。

記憶咭包裝

記憶咭的體積優點在細小以利便携

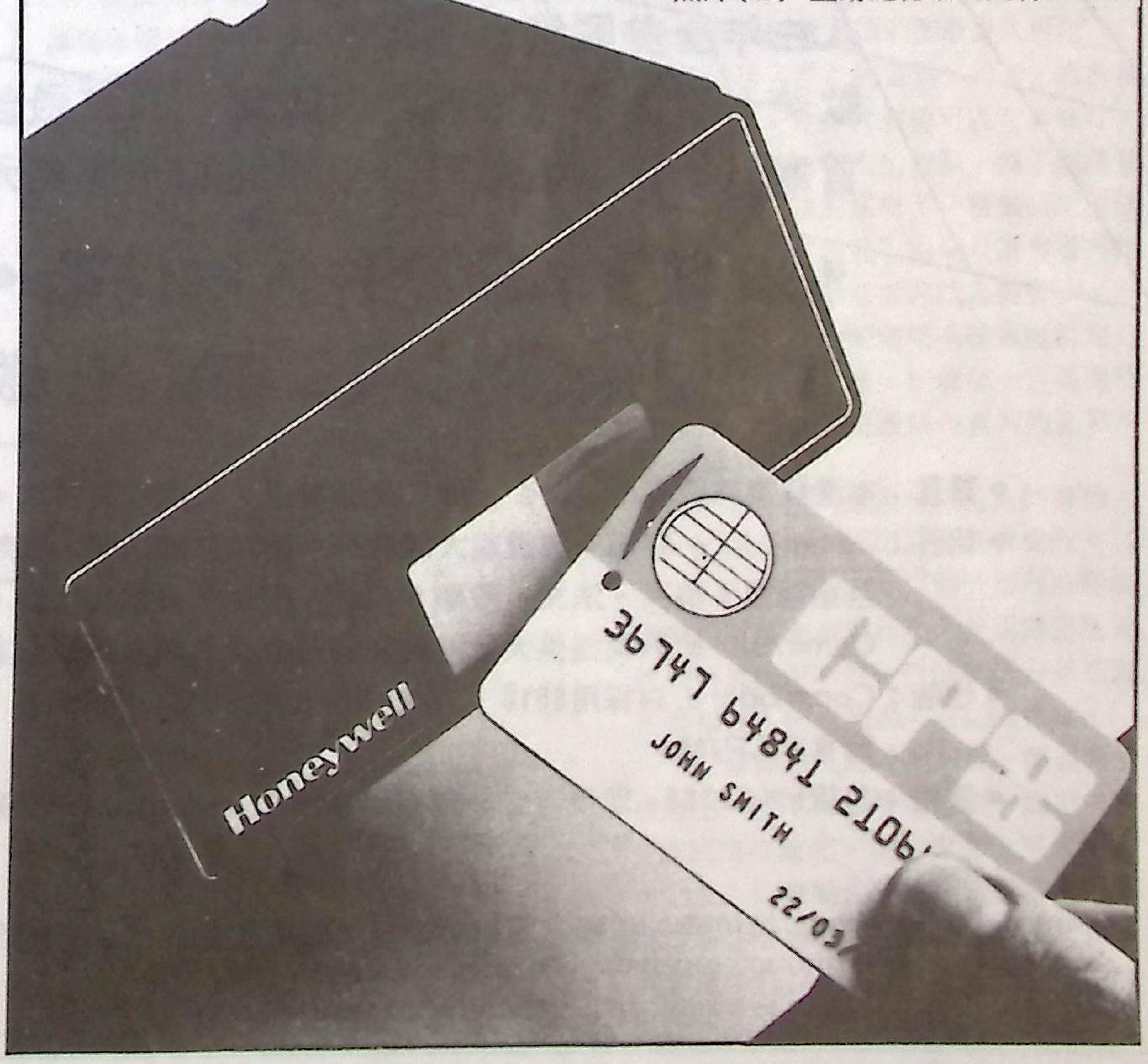
大約五年前，法國電訊局開始研究記憶咭及有關產品。記憶咭初時只着眼用於探索集成電路科技，令如信用咭形狀的物件當作電子貨幣過戶，減少寫支票支付賬目的麻煩及人力虛耗。目前，美國及法國數個試驗點正研究將此概念用於傳視系統。

記憶咭已用作公共電話付款，減少要帶備硬幣的麻煩，消除了錢箱被"爆格"機會，因爲打電話已不用入硬幣。目前市場試驗要解決的難題是記憶咭與交易終端機接觸的可靠性和記憶咭的保密問題。除非記憶咭贏得信任，記憶咭作爲電子貨幣過戶才能夠全面廣泛推行。相信，要數年後，記憶咭始會通行。

美國農業部短期內批出投標供應記憶咭系統替代糧食印花計劃的紙劵。記憶咭有貨幣的相等價值，減少文件、印刷，以處理八百萬戶家庭的印花。

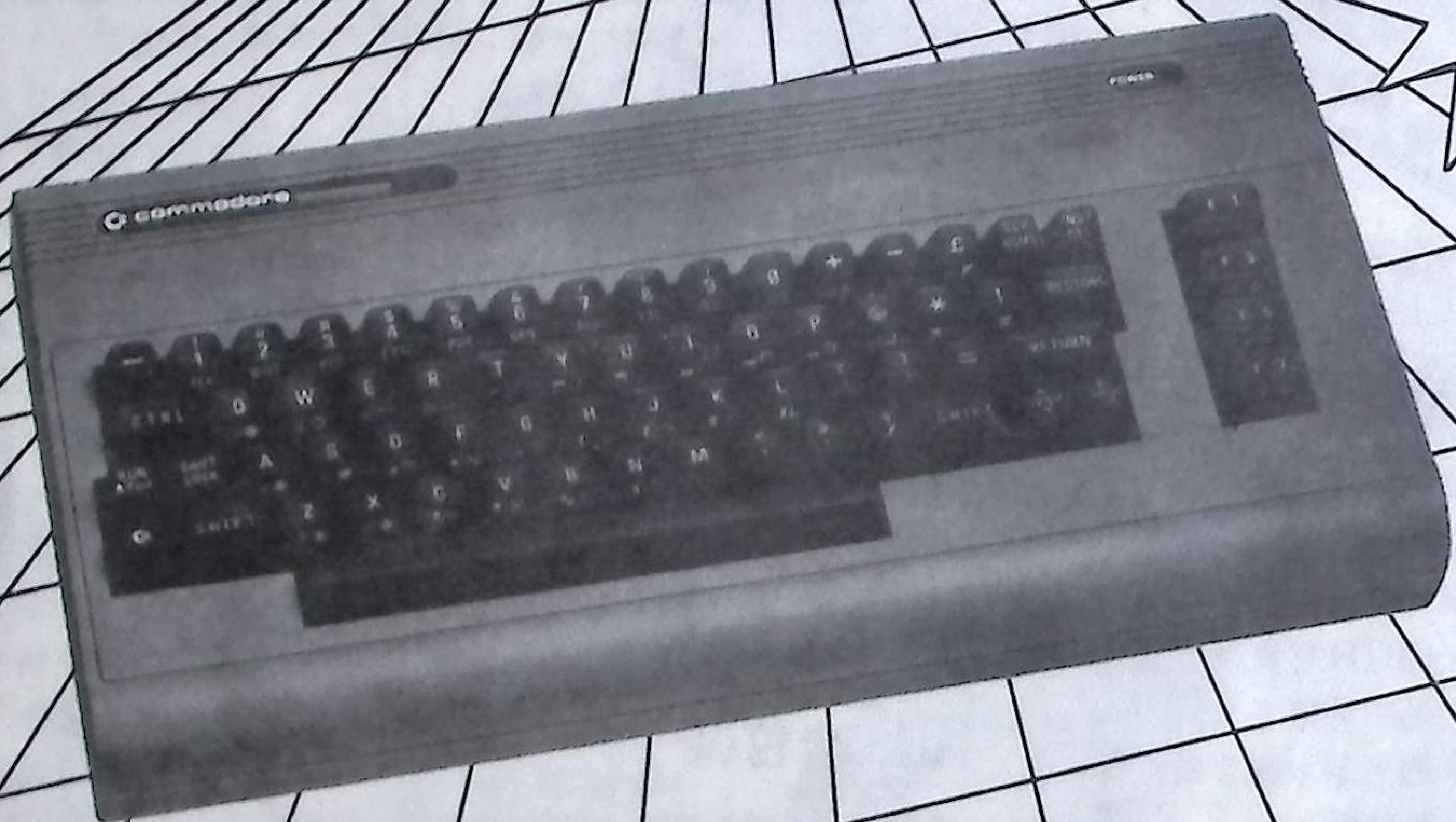
美國軍部亦採用記憶咭以紀錄軍人姓名、階級、受訓及醫療紀錄等等。

照片（2）主動記憶咭的交易終端機



全球最佳家庭電腦
隆重在港推出
COMMODORE 64

程式 600 個
免費贈送



八三年全世界銷量冠軍※
八三年全美國銷量冠軍※
歐美七家著名電腦雜誌評選爲「八三最佳家庭電腦」
首家微電腦製造廠年營業額超越十億美元

•論質量•論軟件•論性能•論價格

COMMODORE 64 皆傲視同羣

- 質量：本港發售的Commodore 64均在英國製造，質量非常可靠。
- 軟件：Commodore 64擁有數量龐大的軟件。例如教育方面，有超過六百種不同程序，內容包括電腦、英文、法文、數學、地理、歷史、自然科學等，全部免費提供用戶抄錄。Commodore 64還提供大量遊戲和商用軟件，而且種類日益增多。
- 性能：Commodore 64採用6510 CPU, 64KB RAM, 16種顏色，並可選用近十種不同電腦語言和CP/M。
- 價格：只賣HK$2388，您便可立刻在家中享用Commodore 64帶給您的各種好處。

總代理：鷹達企業有限公司　灣仔洛克道200號東新商業中心17樓　電話：5-8934447

※售價在US$500以下的微型電腦

本港亦計劃使用記憶咭

本港目前亦有類似的記憶咭計劃使用中。這款記憶咭外型一如一般信用咭，但附有集成電路，採用塑膠製造。

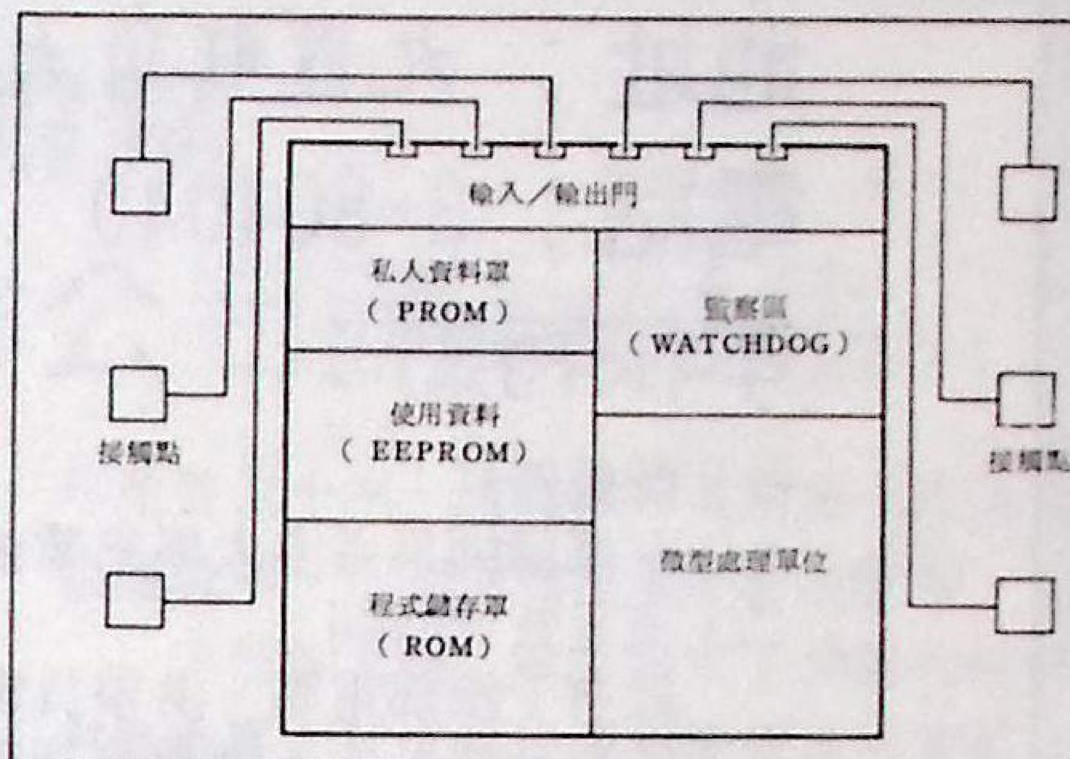
新膠咭，厚度如一般信用咭僅一點二毫米，但嵌有微晶片及供輸入、輸出的電路版，使成微處理器，而具有十六K之記憶容積。

該最新「晶片膠咭」較目前使用的磁帶膠咭，能夠記存更多的數據，和能夠記存數據記錄之變更。

該種膠咭具儲值用途，每次交易後之轉帳均可由儲值帳項內直接減除，此外，該類膠咭因能記錄較多的資訊，故可用作身份證明文件或護照之用。是款膠咭之識別器與磁帶膠咭識別器相若。

若干先進國經已開始率先使用此種晶片膠咭，用以代替現金，而此亦是該等國家準備踏入所謂的『無現金社會』重要的一步。

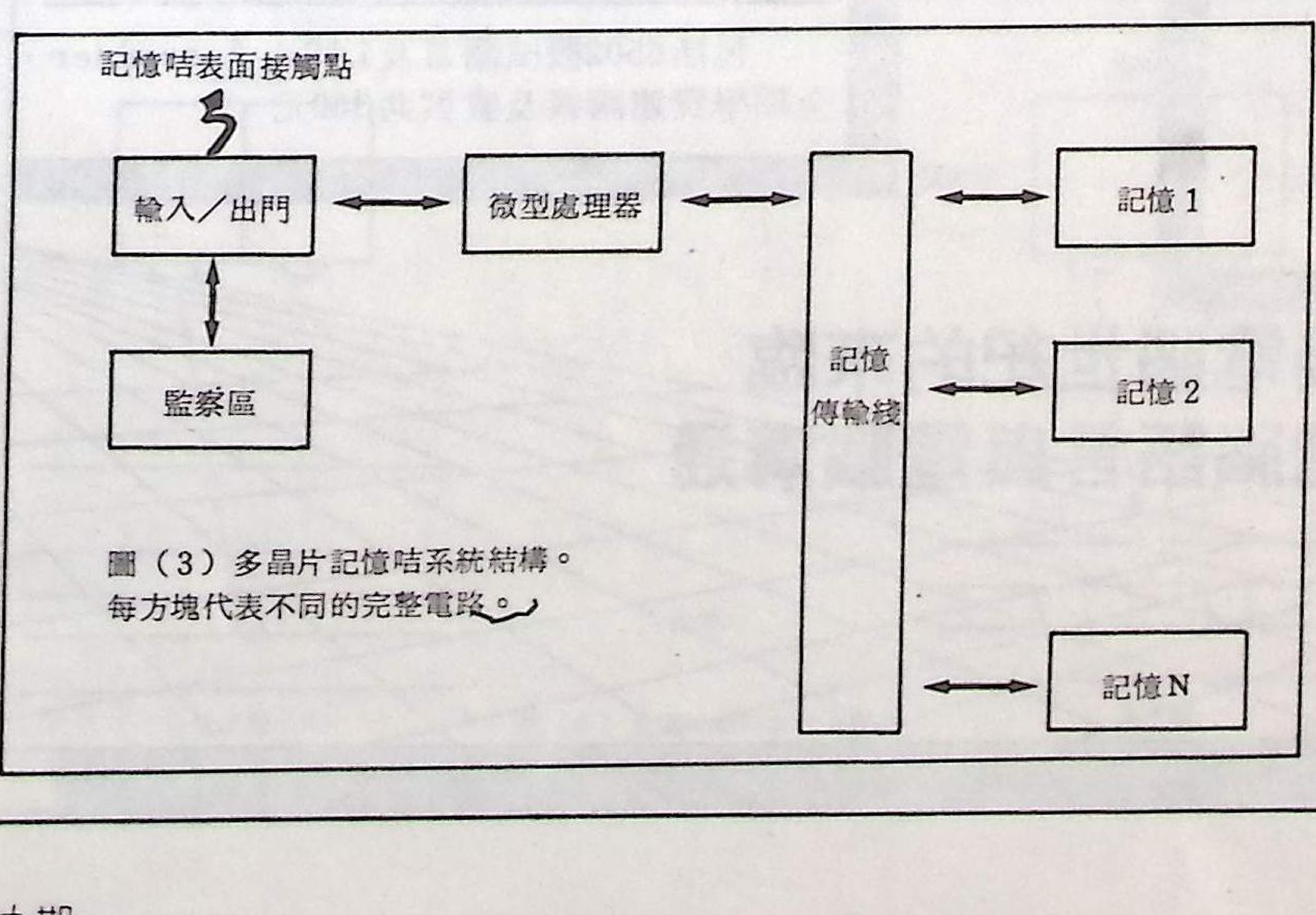
晶片膠咭採用特別的凸印技術，此舉乃有助於數據之安全保密，故此特別適用於財務方面之應用，可用以代替銀行的存摺和支票，或是控股之數據等等；另外，基於該咭能夠記存大量資訊，故可用於高密度之數據記錄，如病歷記錄、個人履歷、酒店登記、信用咭系統及其他種種可預見之用途等。



圖（2）單晶片記憶咭系統的結構

帶。通行的包裝是信用咭的款式，不過，在百分之三吋厚的塑膠層內裝置電子綫路很可能出現毛病。製作塑膠咭時所產生的腐蝕性氣體足以影响整個電路，再者它亦不能用傳統性的半導體纖維技術。

幸運地，自動磁帶搭接（簡稱TAB）技術的出現，解決了這些難題，此技術目前流行應用於電子錶及非常薄的袋裝計算機，TAB 始用於一九六九年，利用彈性印刷電路負荷磁帶（ PRIN—TED—CIVCUIT CARRIER TAPE ）令 IC 搭接（見照片3）。



圖（3）多晶片記憶咭系統結構。
每方塊代表不同的完整電路。

記憶咭系統價格

記憶咭的成本視乎兩個因素：採用的科技及生產規模。生產量高，光條咭價錢每張只五角美金，單晶片咭每張四元五角美金，多晶片咭則是每張八元美金。如果生產量低，價錢就會較爲昂貴一些（見表2）。

除了記憶咭的製作及銷售成本外，還要考慮交易終端機的價格。爲主動記憶咭而設的簡單終端機，價錢可以低至二百五十美元，但被動記憶咭的終端機價錢最少要一千五百美元。這方面的支出是非常可觀的，因爲每一個使用記憶咭的人都要有一部閱讀器。

保密

如果想記憶咭內資料更能保密，安全不會外洩，代價上肯定會水漲船高。例如，記憶咭內存有銀行帳目資料，那麼保密性的需要一定高於儲值車票記憶咭。資料的需要保密程度視乎個人的判斷，它是如何的“私有”。今日通行的咭系統（例如：紙及磁帶咭）只能提供很少的隱私權，但記憶咭則可提供幾個途徑令你保護敏感性的資訊，擁有很高的隱私權。

記憶咭的保密須限制到最少人可以接觸到咭內儲存的資料。再者，容許接觸咭的人，並非代表他可以有權修訂、更改，或增加咭內的資訊，爲了保護咭內的資料，特別採用了一種數據紀錄技術。密碼的“鎖匙”由咭及擁有者共同掌握。只知道鎖匙並非代表擁有權，一個安全保密性的記憶咭系統須能鑑定合法的擁有者。最後，記憶咭一定要保留曾經試圖查詢或更改咭內資料的名單。

鑑證身份

四個方法可以鑑定使用人的身份。（一）咭失去或被偷竊後便會失效。（二)持有人另有密碼。目前的銀行自動櫃員機的磁咭都多用此法，此兩種方法易於應用及費用不昂，但缺點是一旦咭與密碼都被人偷竊知悉，就毫無保障。

另兩個方法則較佳。第一個是利用生理上一些不可改變的特徵，如手指模、聲綫或心電圖，使鑑定身份完美準確。它最大的缺點是操作裝置成本高昂。第二個方法是利用習慣及個人的特點，例如是步行聲及簽署。此方法非常獨特，差不多完全避免遭僞造，而且優點是

資料儲存媒介——記憶咭

操作費不昂貴。利用此種鑑定技巧，可以審查出任何曾經試圖接觸咭內資料的人。

記憶咭同樣如信用咭受到外間欺詐的威脅——如盜用有效的咭，將咭改頭換面，及僞造假咭。這些存在性的威脅在表3a及3b作一比較。雖然記憶咭仍有被不適當使用及遺失的可能性，但它的保密措施仍較磁咭爲優勝。

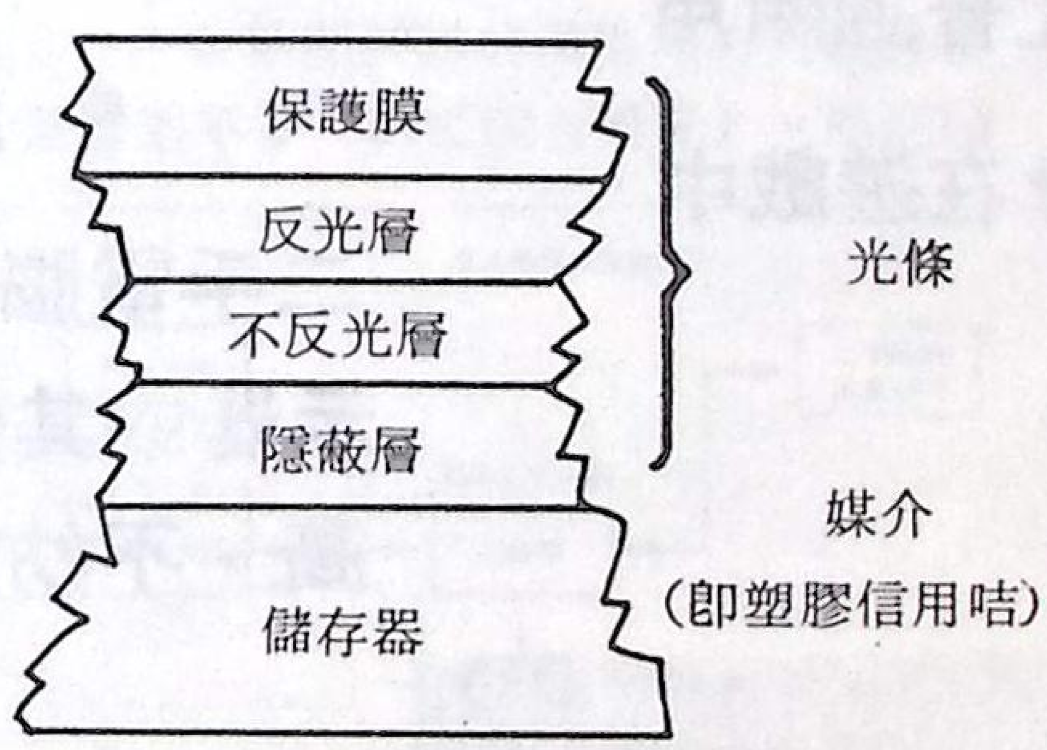
研究此種科技的人聲言記憶咭的保密是毫無疑問的，但直至目前，仍無有力實在的證據支持。近年電腦罪案層出不窮，很多專家對此聲言均置疑。記憶咭系統是比磁帶咭較爲安全，但它實際廣泛使用時的保密程度仍待考驗。目前美國及法國已進行市場試驗。至現在仍未能絕對地解答了保密的疑問。

人爲因素

電腦應用的人爲因素的重要性已日益受到重視，由於記憶咭是細少輕盈隨身携帶的電腦工具，故此必須考慮人爲因素，以確保在某些應用上可以接受使用記憶咭。

目前使用記憶咭的差不多全是金融財務機構，將信用咭改用記憶咭已成趨勢，以減少蒙騙，欺詐及濫用。

傳統的信用咭之所以那麼通行，是因爲它有幾項優點：易於使用，持有人毋須特別的知識，以及生產成本廉宜（所以差不多是免費發出）。但信用咭的最大缺點，是被偷竊後難以追查和易於僞造。不過，記憶咭的優點可以補救信用咭的缺點。記憶咭亦可以同時用作信用咭、存摺、支票簿及自動櫃員咭，集多種用途於一咭。它甚至儲存退休帳目，自動貸款及債券的資料。



圖（4）光條記憶咭的橫切面

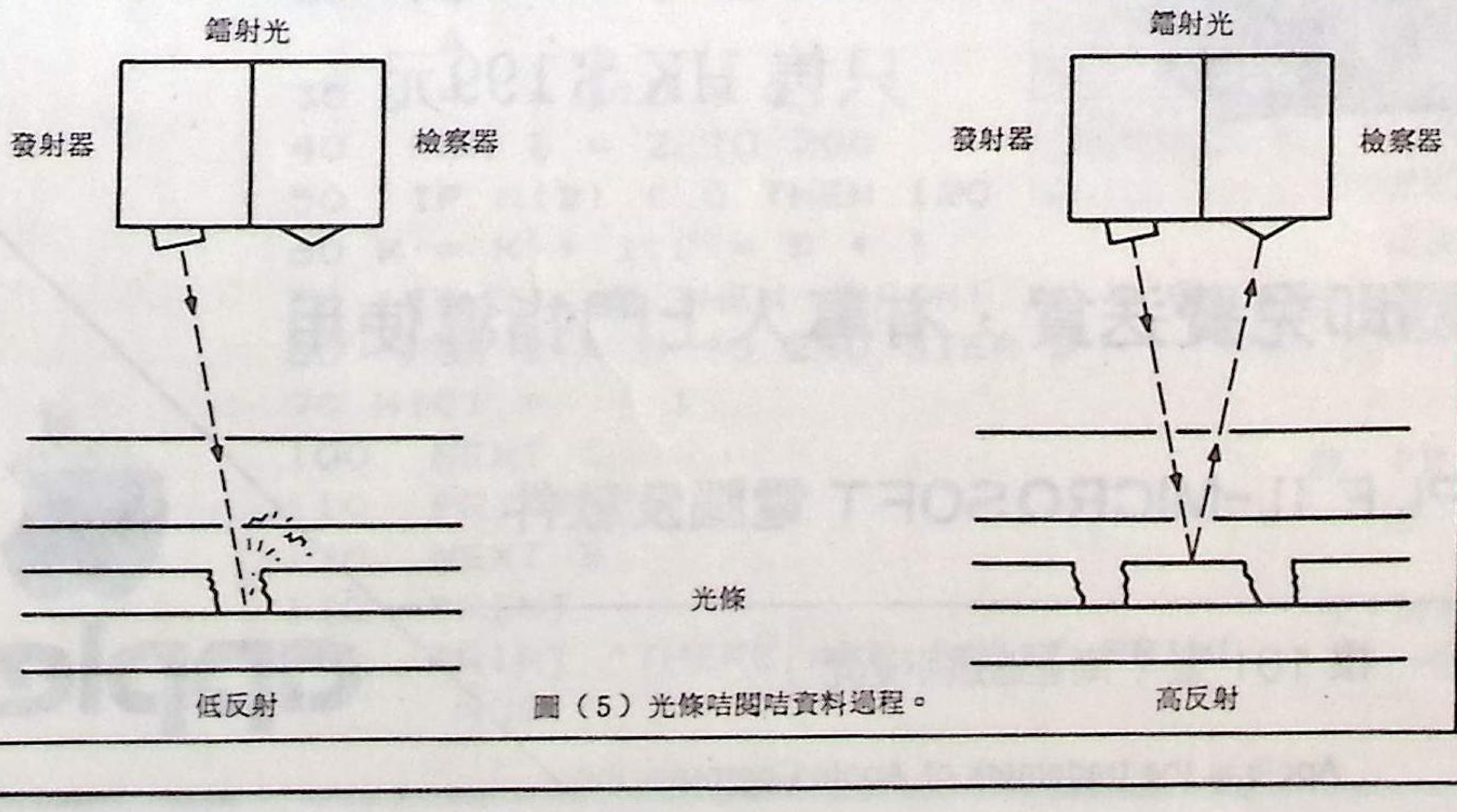
爲了廣大的使用者能接受記憶咭，它必須有某些特點：細小、輕盈及耐久使用。而形狀及咭面設計獨特，就算在微弱的燈光下，仍可分辨出咭的前後左右。另外，終端機亦要輕巧，主動記憶咭的閱讀裝置應以利便爲主，可以隨時修改咭內的儲存資料。

非金融機構應用記憶咭其一例子是用於私人電腦上。私人電腦如任何機器，需要維修。但它不同於機器是能聰明地遮掩毛病。然而採用記憶咭，可以監察電腦的操作，迅速找出及補救毛病。記憶咭會追踪記憶錯誤，電腦內的溫度和濕度，及有邏輯分析功能。

這種電腦毛病診斷記憶咭，大大減少了維修費用，令持有人節省大量時間、金錢及人力。圖6顯示記憶咭的診斷功能。私人電腦使用者倘懷疑有問題出現，可以按下"PANIC"鍵鈕，提醒記憶咭將所有錯誤出現，記憶咭就會紀錄系統內所有不同的物質上及邏輯上的情況。這種診斷能力很有用，特別是私人電腦日趨複雜，這種功能更成爲必需。

裝載軟件

私人電腦軟件有兩項問題：媒體的可靠性及隱私權，大多數的私人電腦軟件是磁性紀錄媒體的軟碟。資料交換讀、寫頭接觸軟碟所產生的磨擦，會令軟碟受到損毁，加上塵埃，手指模，及其他的污染更令損毁加快。但用記憶咭則沒有此問題，所以比軟碟更爲可靠耐用。



圖（5）光條咭閱咭資料過程。

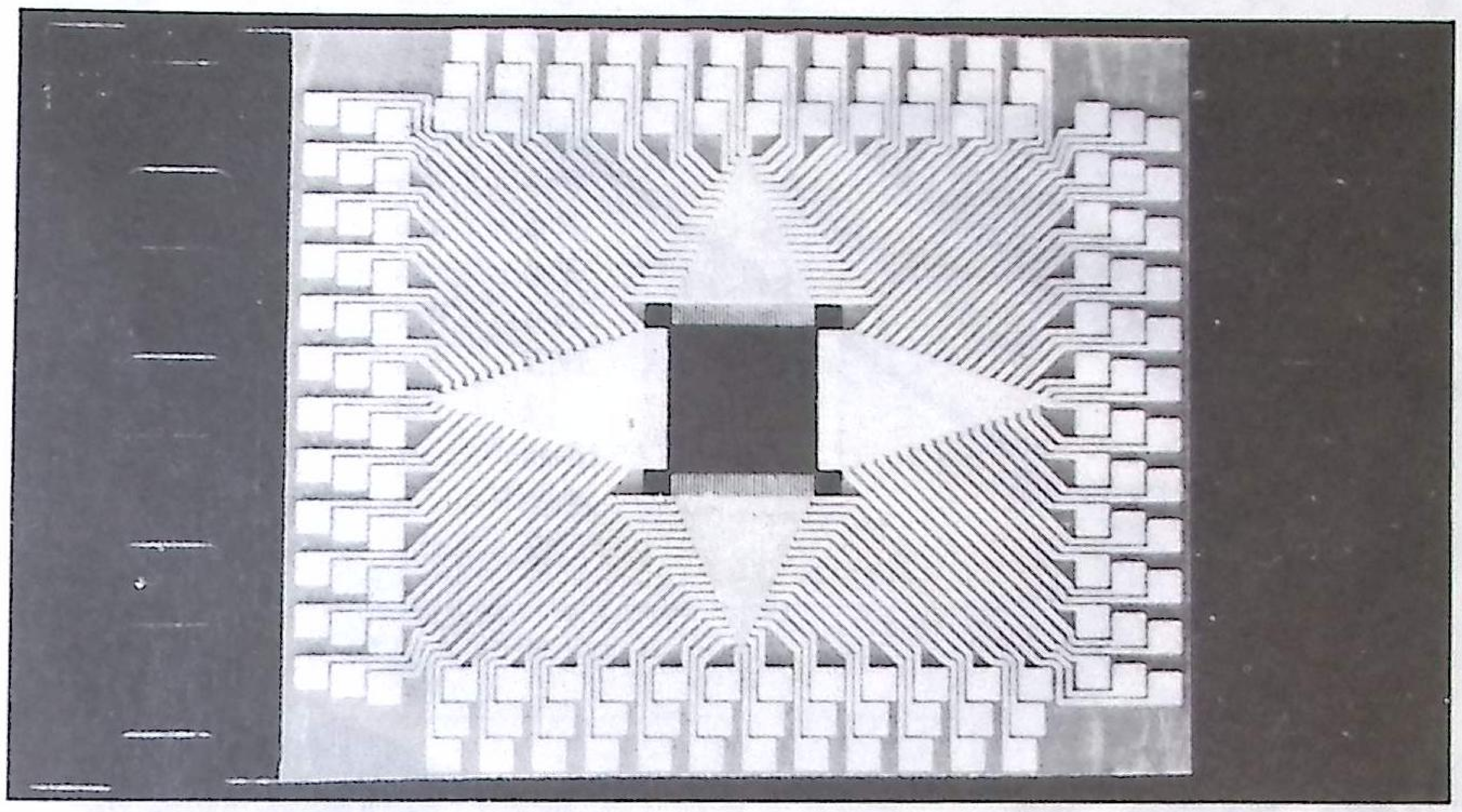
滙豐提款咭新服務 可作公司購物結帳

香港滙豐銀行提款咭持有人可於本年底享用銀行新增的一種服務，在港九多間百貨公司及超級市場購物後，利用提款咭即時結賬，進一步減低携帶大量現款的危險。

滙豐銀行市務部經理方桂生表示，雖然以提款咭結賬，客戶未能享受月結的優待，但却無須另付服務費，只須每年繳費五十年。現時該行共發出一百萬張提款咭，他相信此項服務會受歡迎。

方桂生又稱：歐美國家十分流行的家庭電腦銀行，可於兩年後在本港實現，客戶可以花費一千元購買特別零件，安裝於電視，並可利用電話與銀行電腦通訊，透過電視及電腦購物、付款和處理一切銀行業務。

資料儲存媒介——記憶咭



照片(3) TAB 電路圖

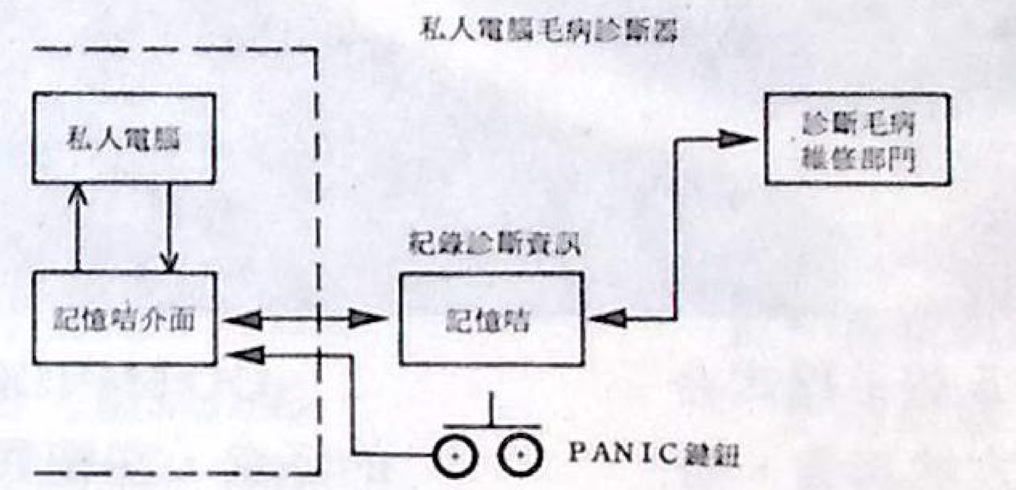
記憶咭種類	低容量	高容量	記憶容量	每字符價格
單晶片	$16.00	$4.50	4K	3.9¢（美元下同）
多晶片	50.00	8.00	16K	3.05¢
光條	4.00	.50	2M	.0002¢
磁條	3.00	.60	212	13.6¢
光／磁兩用條記憶咭	8.00	1.10	2M	.0004¢

圖（2）不同種類記憶咭生產價格表格比較

記憶咭亦保護軟件的隱私權。通常，私人電腦需要在不同磁碟內保存數個程式的副本，以備操作有錯誤或媒體受損毀時，可以補上使用。但副本太多，就有被盜用及洩漏的危險。記憶咭除了不易損壞時，內存的資料亦不容易被刪改。

記憶咭可以儲存的資訊大約是單面5¼吋軟碟容量的二十分之一。科技不斷革新，此缺點相信不久可以克服。

衆數咭閱讀器的價格大約與軟碟驅動器差不多（約是四百美元）。咭閱讀器可以同時處理多個程式，而且提高速度及可靠性。



圖（6）主動記憶咭的方塊圖，用在私人電腦的電路版上。

晶片記憶咭並非是唯一可以儲存資訊的記憶咭。光條記憶咭每張可儲存四百萬個位元，可容許儲存大型程式或大量的小型程式，令私人電腦增加應用途徑。用光條咭，書籍、字典、百科全書及參考書等等都能以機器閱讀。光條記憶咭閱讀器比晶片記憶咭閱讀器爲貴，不過若大量生產，價格預料會下跌至四百美元至八百美元。閱讀器的體積會如5¼吋軟碟驅動碟相若。

記憶咭科技，現雖處於孩提階段，但未來大多數的人會逐漸採用。用途上包括保密、金融事務、資料儲存及私人電腦毛病判斷。至目前，主動記憶咭與被動記憶咭，那種成爲未來主流雖未可知，但記憶咭變成日常生活一部份實無需置疑。□

編印質數答案

```
10 HOME : DIM N(200)
20 FOR A = 1 TO 200:N(A) = A: NEXT
    A
30 K =  - 1:D = 0
40  FOR B = 2 TO 200
50  IF N(B) < 0 THEN 120
60 K = K + 1:D = D + 1
70  IF K = 5 THEN  PRINT :K = 0
80  FOR C = B TO 200 STEP B
90 N(C) =  - 1
100  NEXT C
110  PRINT B;" ";
120  NEXT B
130  PRINT
140  PRINT "THERE ARE ";D;" PRIME
     NOS."
```

註：加多一個 COUNTER，一個 CO— UNT 位數，一個 COUNT PRIME NOS. 數目，K=－1 是使頭一次眞正數0～4。

註：修正上60的遺漏。

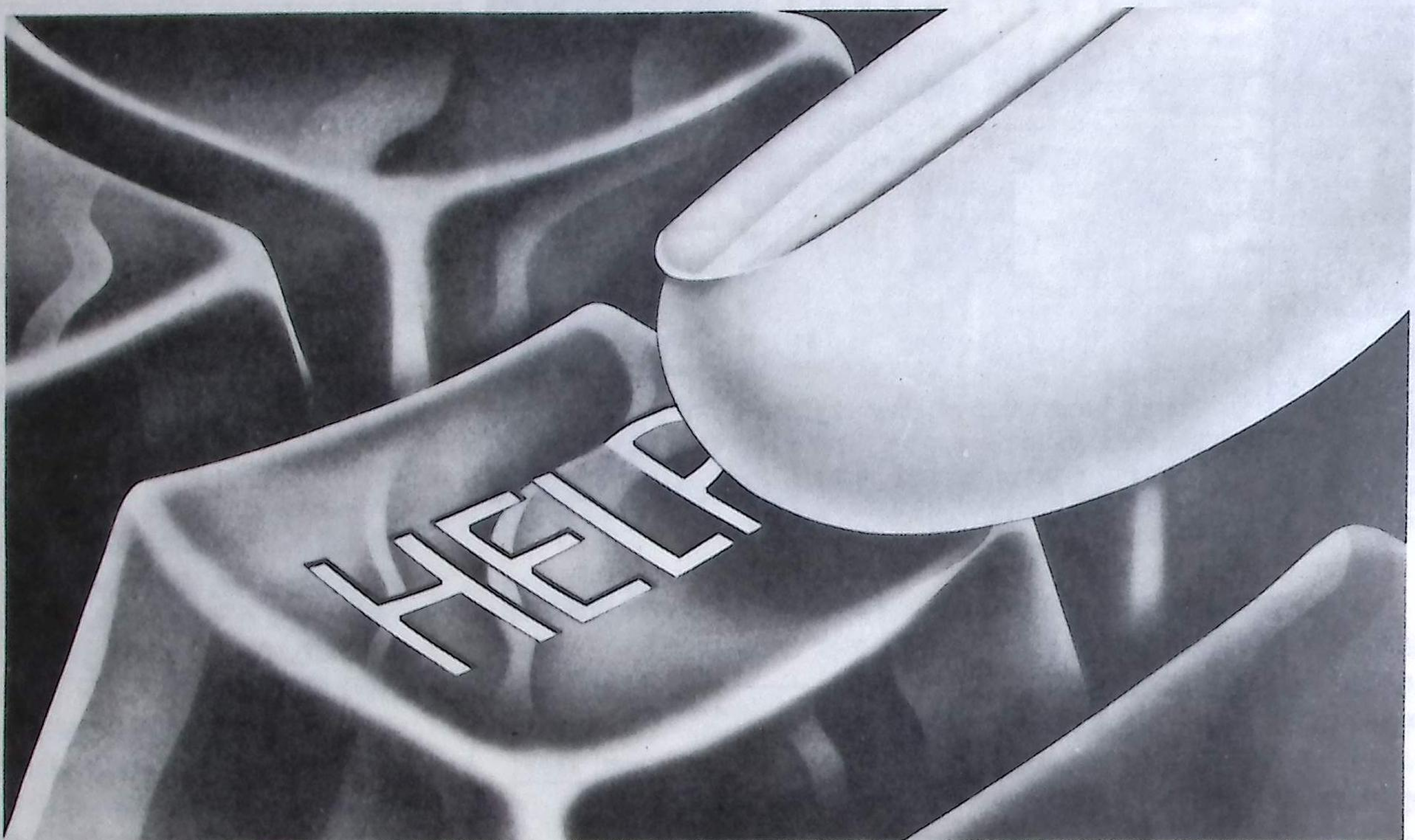
註：PRINT B 敍述在 NEXT C 後，修改上程式印數次刻變數B。

註：開新行，因之前的 PRINT 後有分號。

□

增强鍵盤輸入及螢幕輸出功能的系統

COMPUKEYSOFT



COMPUKEYSOFT 是包括了一系列由5個子程式合成的系統，它將鍵盤輸入及螢幕的輸出功能大大地加強，而且它將 BASIC 及 DOS 運作能力加強，充份發揮64K記憶的優點。

當 COMPUKEYSOFT系統載入記憶時，閣下可以掌握以下的功能：

①無論何時何地皆可使用大小階功能，不論在 BASIC，DOS 或監督程式系統中，令 BASIC 能分辨以小階字爲名的變數，而且擴展了輸出網路，令 80 COLUMN 咭亦能發揮功用。

②建立特別的 I／O網路，當作40字行顯示時，會以閃爍的光點代替正常情形下的光點，以分辨 COMPUKEYSO－FT 系統在使用中。

③一個被改良過的 BASIC 翻譯網路使命令能同時以大小階印出，這個功能使程式在列印出來時更加專業化。

④將 DOS 3.3 MASTER 碟上的 FPBASIC載往RA－MCARD上第二組4K BANK 上，使電影系統仍保留48K的正常使用空間，不會因此而減少。

⑤內置了編輯功能，這個功能是以新加入的命令 EDIT 來控制，閣下可以在程式行句與行句之間加插或刪除一些字元， 這些字元若非使用特殊的工具程式時是無法加入或刪除的。

系統的使用

COMPUKEYSOFT 系統充份發揮80行咭在編輯功能的好處，在視覺上會舒適得多，它的操作需要將80行咭插在 SLOT 3上，目的是使電腦在運行 CP／M及PASCAL時不需再將80行咭搬移。

要使本系統工作只需 RUN COMPUKEYSOFT H－ELLO ，它會將一系列的機械語言程式載入記憶裡，當完成後便將控制權交回。

在這裡需要提醒各位， COMPUKEYSOFT系統執行前需要在 SLOT 0上插上16K RAM 咭，否則系統是不能運行的，而且最好有80行字咭插入在 SLOT 3 ，沒有的話則不能以80行顯示。在DOS 系統方面需要是 DOS 3.3或與它相容的系統。

操控鍵簡介

啓動本系統，先鍵入 RUN COMPUKEYSOFT HELLO ，等候一系列的副程式載入。完成這步後你將被置於 COMPUKEYSOFT 系統的正常狀態，即大階操作模式之下，只需按下〔ESC〕〔L〕便可以選擇以小階輸入命令了，選擇以大階輸入時，按〔ESC〕〔U〕便可以。

除了以上兩種輸入模式之外，尙有另外兩種新模式。

① NEXT 狀態──如需要輸入一些東西入程式內而該程式必須以正常之鍵盤輸入，則按〔ESC〕〔N〕（輸入字元）這個功能可令本系統的輸入網路中斷，使鍵盤功能回復到一般情況。

Chr	Key	Chr	Key	Chr	Key	Chr	Key	Chr	Key	Chr	Key	Chr	Key	Chr	Key
nul	cs-P	dle	c-P	spc	spc	0	0	@	c-P	P	s-p	‘	E’	p	p
soh	c-A	dc1	c-Q	!	!	1	1	A	s-a	Q	s-q	a	a	q	q
stx	c-B	dc2	c-R	"	"	2	2	B	s-b	R	s-r	b	b	r	r
etx	c-C	dc3	c-S	#	#	3	3	C	s-c	S	s-s	c	c	s	s
eot	c-D	dc4	c-T	$	$	4	4	D	s-d	T	s-t	d	d	t	t
enq	c-E	nak	c-U	%	%	5	5	E	s-e	U	s-u	e	e	u	u
ack	c-F	syn	c-V	&	&	6	6	F	s-f	V	s-v	f	f	v	v
bel	c-G	etb	c-W	’	’	7	7	G	s-g	W	s-w	g	g	w	w
bs	c-H	can	c-X	(	(	8	8	H	s-h	X	s-x	h	h	x	x
ht	c-I	em	c-Y	)	)	9	9	I	s-i	Y	s-y	i	i	y	y
lf	c-J	sub	c-Z	*	*	:	:	J	s-j	Z	s-z	j	j	z	z
vt	c-K	esc	ENE	+	+	;	;	K	s-k	[	E1	k	k	{	E6
ff	c-L	fs	E,	,	,	<	<	L	s-l	\	E2	l	l	|	E7
cr	c-M	gs	cs-M	–	–	=	=	M	s-m	]	E3	m	m	}	E8
so	c-N	rs	cs-N	.	.	>	>	N	s-n	^	E4	n	n	~	E9
si	c-O	us	E/	/	/	?	?	O	s-o	_	E5	o	o	del	E0

c = control, s = shift, cs = control/shift, E = ESCape key

表A

② TICK 狀態——這個狀態的啓動只需按〔ESC〕〔T〕便可工作，它的作用是當你按下ESC 鍵後再按 I，J，K，L或A，B，C，D來移動光點時可以發出一下TICK 的聲音。

③除了以上兩種功能，COMPUKEYSOFT尚可輸入一些特殊字元，例如按〔ESC〕〔5〕可產生一個 UNDERLINE字元，請參考表①。

註：(^) 這個字元可以按〔ESC〕〔4〕或在小階狀態時按〔CTRL〕〔N〕來獲得。

④在 BASIC，MONITOR及DOS 之下可以使用〔ESC〕〔L〕以小階操作的模式，可以鍵進任何程式內容，在本系統運作時可以接受以大小階來控制6502，包括DOS 的檔案名字，並且更重要的是加強了 BASIC 使變數名字能以大小階來命名，而同一變數在大階及小階可存放不同的數值，例如A=10，a=20等。由於這些性能被加入而不令記憶空間降低，因此被逼使卡式機的出入網路被中斷，而不能使用卡式機，相信這一點對於使用磁碟機的人來說是沒有影响的。

⑤本系統中最優異性能—— EDIT 命令這個編輯模式以EDIT 命令來啓動，使 BASIC 程式在除錯及輸入時更有效率，避免了經常輸入 POKE 33，33的麻煩。例如〔EDIT〕〔100〕可以編輯程式的第100行，它首先清除螢幕，並將第100 行顯示在頂端部份（80行咭亦相同）。

當 EDIT 命令被啓動時，被隱藏在程式之內的控制字元會以反白字元顯示，並且用盡40行顯示，把不要的空白行刪掉。

EDIT 命令可以後接例如：

A、100—200 或 100，200 使第100至200行進入編輯模式之下。

B、EDIT 命令更可以用來找尋任何程式內字元之用，字元的長度更可達64個字長。例如 EDIT〔HELLO〕會將全個程式內的 HELLO 找出來，並進行編輯狀態以便修改。

C、另外〔—〕這個UNDERLINE 字元（由按入〔ESC〕〔5〕產生），可以用作爲任何字元被取代之用，即它是一個WILDCARD字元，例如 EDIT "05—_ _—84" 會把程式內的任何日期找出來，因此會很實用。

D、在編輯狀態下，光點的移動將由 CTRL，W，A，S，Z 來控制上，左，右，下 4 個方向，而不是用 ESC 鍵。因爲W，A，S，Z 被排列成菱形在鍵盤上的左方，因此使用上會很方便。

E、在編輯時，CTRL—D，I，N，T 及X發揮很多正常狀態下不能做到的事，例如：

CTRL—D 將光點停留在位址上字元刪去。

CTRL—I 在光點停留位址開始加插任何字元式或字句往程式行裏，當加插完畢後按一次←鍵可以關掉插入狀態。

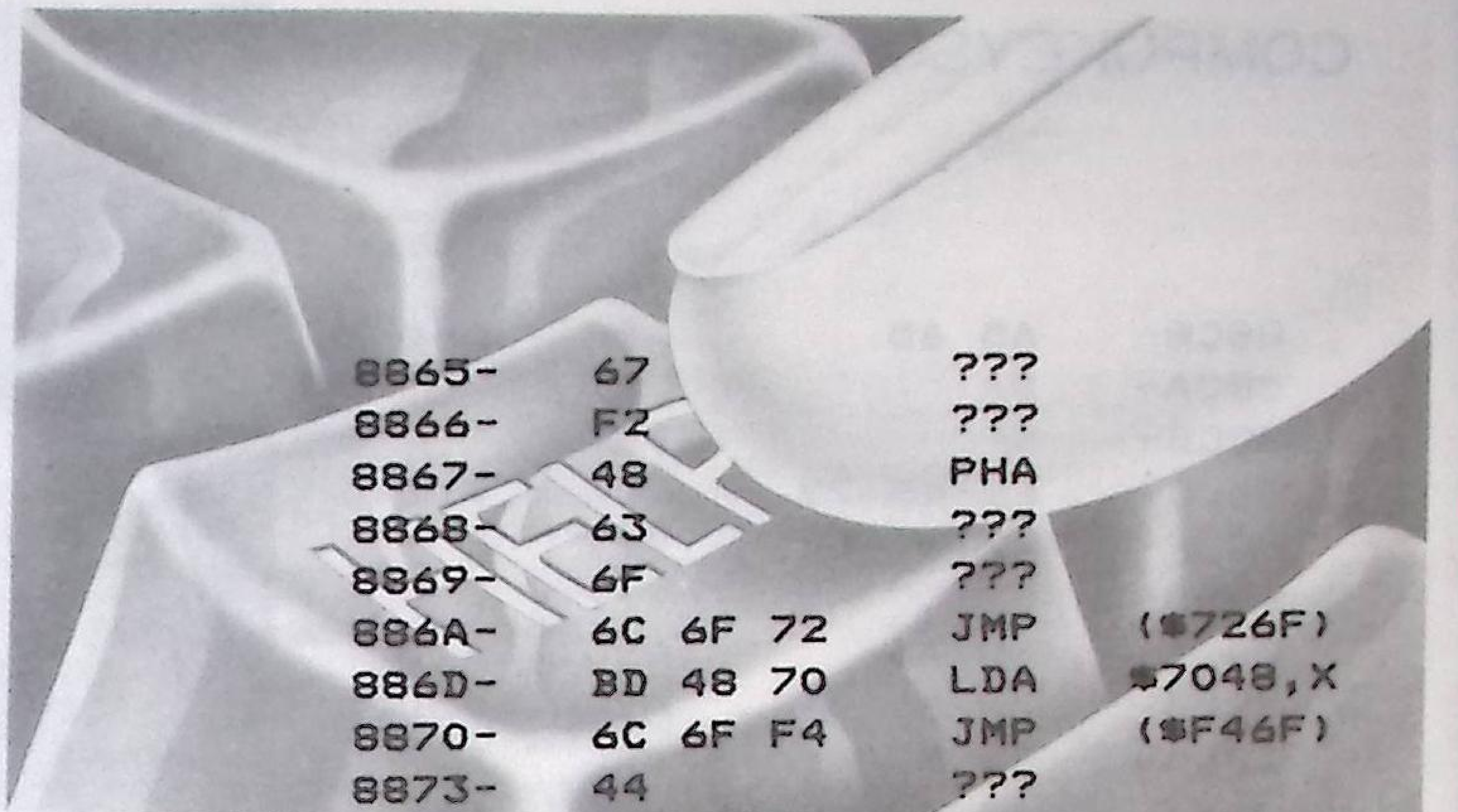
CTRL—N 這個命令和〔ESC〕〔N〕命令有此相似，CTRL—N 可以令你進入第一種模式，即 NEXT 模式之下，不過 CTRL—N 是在編輯模式下呼叫 NEXT 模式，這在你希望加入一些在普通情況下不能加插到程式上的字元尤其有用。例如 CTRL—M（回車）CTRL—H（BLACKSPACE）等控制字元。

CTRL—T 將光點以後的東西刪除。省却按入多次 CTRL—D。

CTRL—X 將編輯過的該行取消而不改變記憶體內的該行內容。

RETURN 不論光點在任何位置，按下〔RETURN〕可以令該行輸入到記憶體取代舊有的一行，而不論該行是否被編輯過。

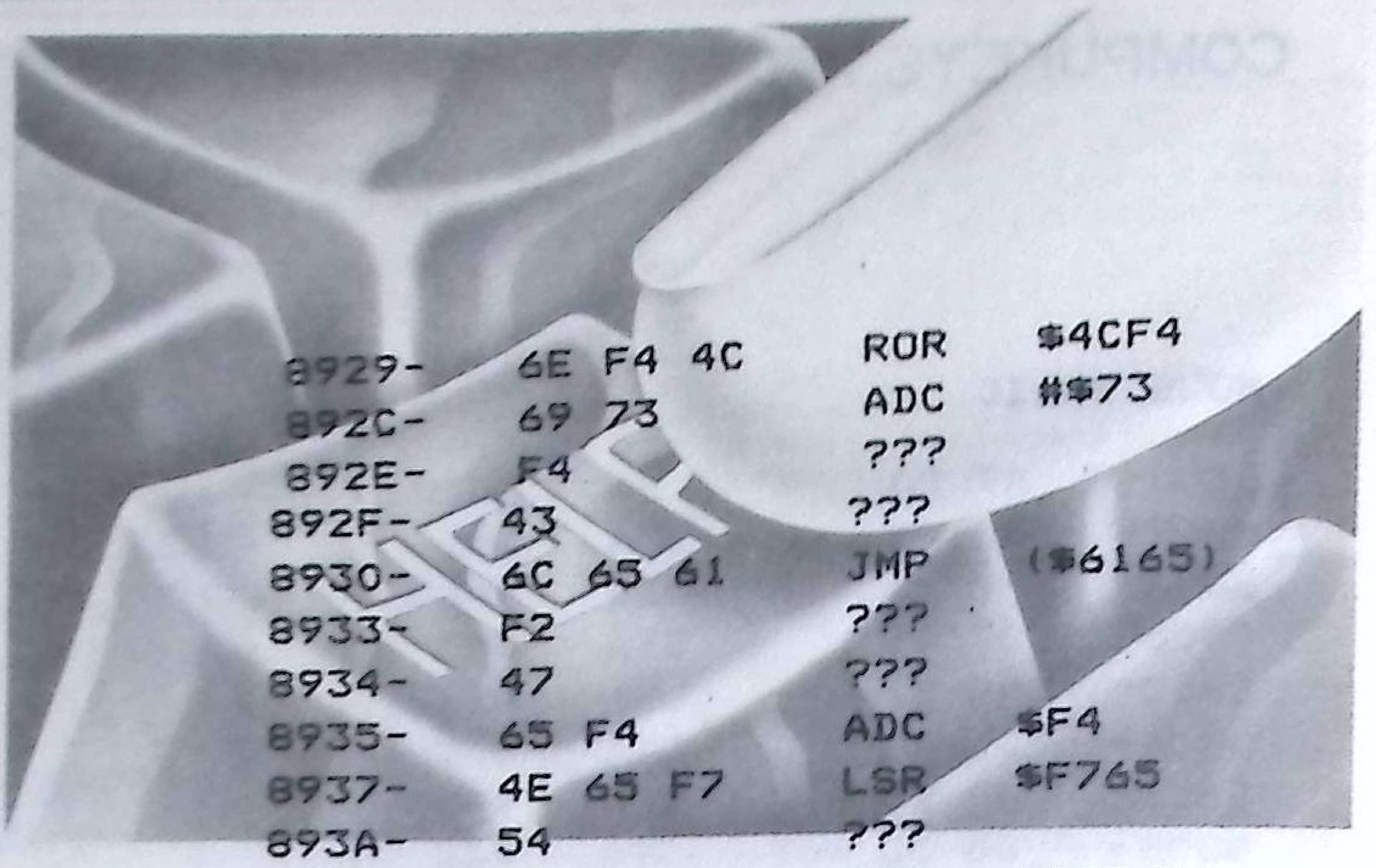
以往若要把一列行號抄往程式內其它地方時，須要 LIST 該行，用ESC 命令移光點到最左，改行號，用→鍵覆複製一遍，若果很多行時便會不勝其煩。現在你只須 EDIT 該列行號，改掉行號後按 RETURN 便可，很方便吧！□



MOVE BASIC

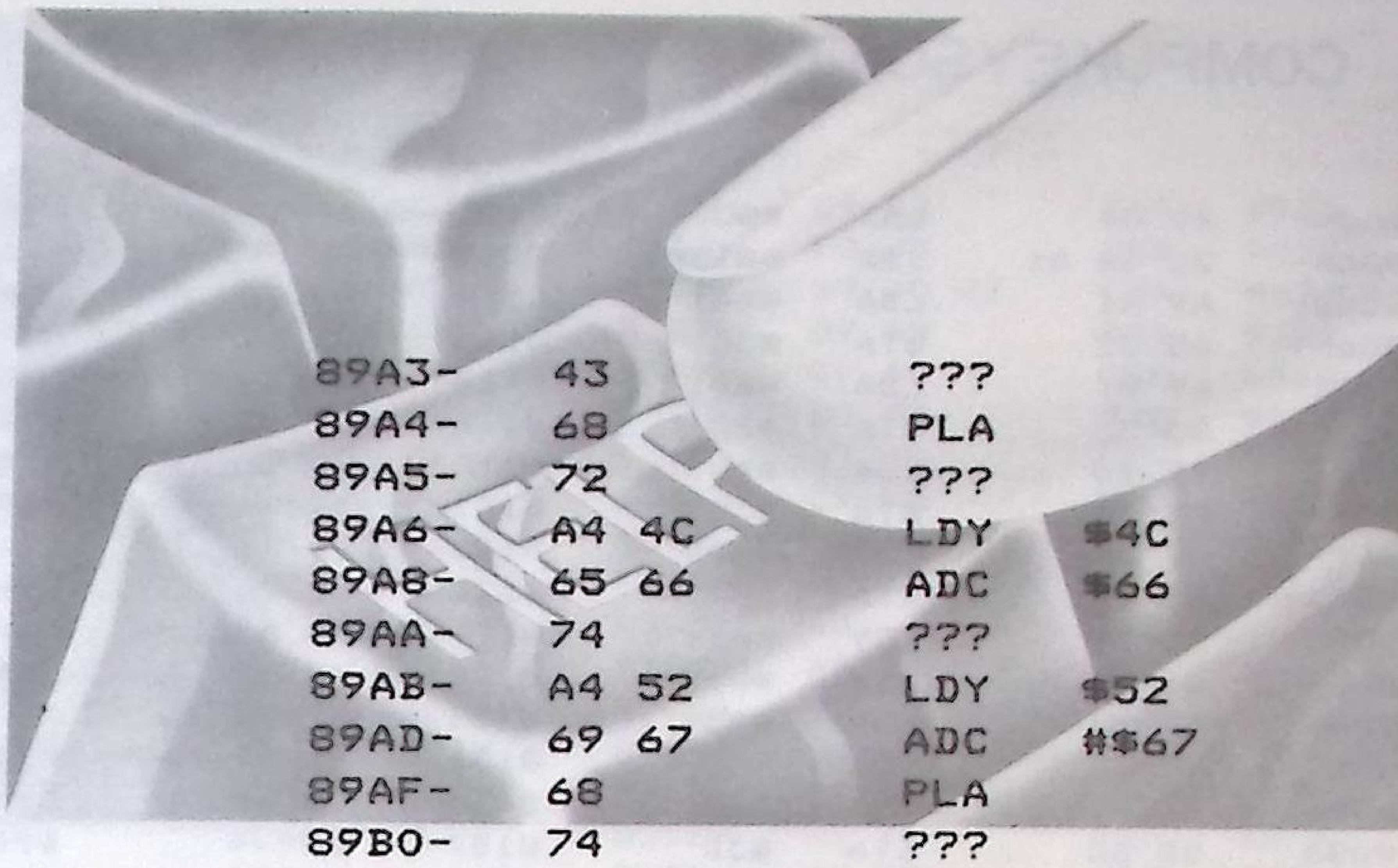
```
8800-    A9 D0       LDA   #$D0
8802-    85 42       STA   $42
8804-    A9 D0       LDA   #$D0
8806-    85 43       STA   $43
8808-    A9 27       LDA   #$27
880A-    85 3C       STA   $3C
880C-    A9 88       LDA   #$88
880E-    85 3D       STA   $3D
8810-    A9 B5       LDA   #$B5
8812-    85 3E       STA   $3E
8814-    A9 89       LDA   #$89
8816-    85 3F       STA   $3F
8818-    2C 81 C0    BIT   $C081
881B-    2C 81 C0    BIT   $C081
881E-    A0 00       LDY   #$00
8820-    20 2C FE    JSR   $FE2C
8823-    2C 82 C0    BIT   $C082
8826-    60          RTS
8827-    45 6E       EOR   $6E
8829-    E4 46       CPX   $46
882B-    6F          ???
882C-    F2          ???
882D-    4E 65 78    LSR   $7865
8830-    F4          ???
8831-    44          ???
8832-    61 74       ADC   ($74,X)
8834-    E1 49       SBC   ($49,X)
8836-    6E 70 75    ROR   $7570
8839-    F4          ???
883A-    44          ???
883B-    65 EC       ADC   $EC
883D-    44          ???
883E-    69 ED       ADC   #$ED
8840-    52          ???
8841-    65 61       ADC   $61
8843-    E4 47       CPX   $47
8845-    F2          ???
8846-    54          ???
8847-    65 78       ADC   $78
8849-    F4          ???
884A-    50 72       BVC   $88BE
884C-    A3          ???
884D-    49 6E       EOR   #$6E
884F-    A3          ???
8850-    43          ???
8851-    61 6C       ADC   ($6C,X)
8853-    EC 50 6C    CPX   $6C50
8856-    6F          ???
8857-    F4          ???
8858-    48          PHA
8859-    6C 69 EE    JMP   ($EE69)
885C-    56 6C       LSR   $6C,X
885E-    69 EE       ADC   #$EE
8860-    48          PHA
8861-    67          ???
8862-    72          ???
8863-    B2          ???
8864-    48          PHA
8865-    67          ???
8866-    F2          ???
8867-    48          PHA
8868-    63          ???
8869-    6F          ???
886A-    6C 6F 72    JMP   ($726F)
886D-    BD 48 70    LDA   $7048,X
8870-    6C 6F F4    JMP   ($F46F)
8873-    44          ???
8874-    72          ???
8875-    61 F7       ADC   ($F7,X)
8877-    58          CLI
8878-    64          ???
8879-    72          ???
887A-    61 F7       ADC   ($F7,X)
887C-    48          PHA
887D-    74          ???
887E-    61 E2       ADC   ($E2,X)
8880-    48          PHA
8881-    6F          ???
8882-    6D E5 52    ADC   $52E5
8885-    6F          ???
8886-    74          ???
8887-    BD 53 63    LDA   $6353,X
888A-    61 6C       ADC   ($6C,X)
888C-    65 BD       ADC   $BD
888E-    53          ???
888F-    68          PLA
8890-    6C 6F 61    JMP   ($616F)
8893-    E4 54       CPX   $54
8895-    72          ???
8896-    61 63       ADC   ($63,X)
8898-    E5 4E       SBC   $4E
889A-    6F          ???
889B-    54          ???
889C-    72          ???
889D-    61 63       ADC   ($63,X)
889F-    E5 4E       SBC   $4E
88A1-    6F          ???
88A2-    72          ???
88A3-    6D 61 EC    ADC   $EC61
88A6-    49 6E       EOR   #$6E
88A8-    76 65       ROR   $65,X
88AA-    72          ???
88AB-    73          ???
88AC-    E5 46       SBC   $46
88AE-    6C 61 73    JMP   ($7361)
88B1-    E8          INX
88B2-    43          ???
88B3-    6F          ???
88B4-    6C 6F 72    JMP   ($726F)
88B7-    BD 50 6F    LDA   $6F50,X
88BA-    F0 56       BEQ   $8912
88BC-    74          ???
88BD-    61 E2       ADC   ($E2,X)
88BF-    48          PHA
88C0-    69 6D       ADC   #$6D
88C2-    65 6D       ADC   $6D
88C4-    BA          TSX
88C5-    4C 6F 6D    JMP   $6D6F
```

COMPUKEYSOFT



```
88C8-    65 6D       ADC   $6D
88CA-    BA          TSX
88CB-    4F          ???
88CC-    6E 45 72    ROR   $7245
88CF-    F2          ???
88D0-    52          ???
88D1-    65 73       ADC   $73
88D3-    75 6D       ADC   $6D,X
88D5-    E5 52       SBC   $52
88D7-    65 63       ADC   $63
88D9-    61 6C       ADC   ($6C,X)
88DB-    EC 53 74    CPX   $7453
88DE-    6F          ???
88DF-    72          ???
88E0-    E5 53       SBC   $53
88E2-    70 65       BVS   $8949
88E4-    65 64       ADC   $64
88E6-    BD 4C 65    LDA   $654C,X
88E9-    F4          ???
88EA-    47          ???
88EB-    6F          ???
88EC-    54          ???
88ED-    EF          ???
88EE-    52          ???
88EF-    75 EE       ADC   $EE,X
88F1-    49 E6       EOR   #$E6
88F3-    52          ???
88F4-    65 73       ADC   $73
88F6-    74          ???
88F7-    6F          ???
88F8-    72          ???
88F9-    E5 A6       SBC   $A6
88FB-    47          ???
88FC-    6F          ???
88FD-    53          ???
88FE-    75 E2       ADC   $E2,X
8900-    52          ???
8901-    65 74       ADC   $74
8903-    75 72       ADC   $72,X
8905-    EE 52 65    INC   $6552
8908-    ED 53 74    SBC   $7453
890B-    6F          ???
890C-    F0 4F       BEQ   $895D
890E-    EE 57 61    INC   $6157
8911-    69 F4       ADC   #$F4
8913-    45 64       EOR   $64
8915-    69 F4       ADC   #$F4
8917-    53          ???
8918-    61 76       ADC   ($76,X)
891A-    E5 44       SBC   $44
891C-    65 E6       ADC   $E6
891E-    50 6F       BVC   $898F
8920-    6B          ???
8921-    E5 50       SBC   $50
8923-    72          ???
8924-    69 6E       ADC   #$6E
8926-    F4          ???
8927-    43          ???
8928-    6F          ???
8929-    6E F4 4C    ROR   $4CF4
892C-    69 73       ADC   #$73
892E-    F4          ???
892F-    43          ???
8930-    6C 65 61    JMP   ($6165)
8933-    F2          ???
8934-    47          ???
8935-    65 F4       ADC   $F4
8937-    4E 65 F7    LSR   $F765
893A-    54          ???
893B-    61 62       ADC   ($62,X)
893D-    A8          TAY
893E-    74          ???
893F-    EF          ???
8940-    46 EE       LSR   $EE
8942-    53          ???
8943-    70 63       BVS   $89A8
8945-    A8          TAY
8946-    74          ???
8947-    68          PLA
8948-    65 EE       ADC   $EE
894A-    61 F4       ADC   ($F4,X)
894C-    6E 6F F4    ROR   $F46F
894F-    73          ???
8950-    74          ???
8951-    65 F0       ADC   $F0
8953-    AB          ???
8954-    AD AA AF    LDA   $AFAA
8957-    DE 61 6E    DEC   $6E61,X
895A-    E4 6F       CPX   $6F
895C-    F2          ???
895D-    BE BD BC    LDX   $BCBD,Y
8960-    53          ???
8961-    67          ???
8962-    EE 49 6E    INC   $6E49
8965-    F4          ???
8966-    41 62       EOR   ($62,X)
8968-    F3          ???
8969-    55 73       EOR   $73,X
896B-    F2          ???
896C-    46 72       LSR   $72
896E-    E5 53       SBC   $53
8970-    63          ???
8971-    72          ???
8972-    6E A8 50    ROR   $50A8
8975-    64          ???
8976-    EC 50 6F    CPX   $6F50
8979-    F3          ???
897A-    53          ???
897B-    71 F2       ADC   ($F2),Y
897D-    52          ???
897E-    6E E4 4C    ROR   $4CE4
8981-    6F          ???
8982-    E7          ???
8983-    45 78       EOR   $78
8985-    F0 43       BEQ   $89CA
8987-    6F          ???
8988-    F3          ???
8989-    53          ???
```

COMPUKEYSOFT



```
898A-    69 EE       ADC   #$EE
898C-    54          ???
898D-    61 EE       ADC   ($EE,X)
898F-    41 74       EOR   ($74,X)
8991-    EE 50 65    INC   $6550
8994-    65 EB       ADC   $EB
8996-    4C 65 EE    JMP   $EE65
8999-    53          ???
899A-    74          ???
899B-    72          ???
899C-    A4 56       LDY   $56
899E-    61 EC       ADC   ($EC,X)
89A0-    41 73       EOR   ($73,X)
89A2-    E3          ???
89A3-    43          ???
89A4-    68          PLA
89A5-    72          ???
89A6-    A4 4C       LDY   $4C
89A8-    65 66       ADC   $66
89AA-    74          ???
89AB-    A4 52       LDY   $52
89AD-    69 67       ADC   #$67
89AF-    68          PLA
89B0-    74          ???
89B1-    A4 4D       LDY   $4D
89B3-    69 64       ADC   #$64
89B5-    A4 00       LDY   $00
```
□

MOVE DOS

```
8700-    A9 DF       LDA   #$DF
8702-    85 42       STA   $42
8704-    A9 BC       LDA   #$BC
8706-    85 43       STA   $43
8708-    A9 46       LDA   #$46
870A-    85 3C       STA   $3C
870C-    A9 87       LDA   #$87
870E-    85 3D       STA   $3D
8710-    A9 5F       LDA   #$5F
8712-    85 3E       STA   $3E
8714-    A9 87       LDA   #$87
8716-    85 3F       STA   $3F
8718-    A0 00       LDY   #$00
871A-    20 2C FE    JSR   $FE2C
871D-    A9 4C       LDA   #$4C
871F-    8D EB 9F    STA   $9FEB
8722-    8D 0E A1    STA   $A10E
8725-    A9 DF       LDA   #$DF
8727-    8D EC 9F    STA   $9FEC
872A-    A9 F1       LDA   #$F1
872C-    8D 0F A1    STA   $A10F
872F-    A9 BC       LDA   #$BC
8731-    8D ED 9F    STA   $9FED
8734-    8D 10 A1    STA   $A110
8737-    A9 EA       LDA   #$EA
8739-    8D EE 9F    STA   $9FEE
873C-    8D EF 9F    STA   $9FEF
873F-    8D 48 A6    STA   $A648
8742-    8D 49 A6    STA   $A649
8745-    60          RTS
8746-    C9 E1       CMP   #$E1
8748-    90 06       BCC   $8750
874A-    C9 FB       CMP   #$FB
874C-    B0 02       BCS   $8750
874E-    29 5F       AND   #$5F
8750-    29 7F       AND   #$7F
8752-    59 84 A8    EOR   $A884,Y
8755-    4C F0 9F    JMP   $9FF0
8758-    29 DF       AND   #$DF
875A-    DD 40 A9    CMP   $A940,X
875D-    4C 11 A1    JMP   $A111
8760-    00          BRK
```

EXTEND BASIC

```
8000-    A9 00       LDA   #$00
8002-    85 42       STA   $42
8004-    A9 D0       LDA   #$D0
8006-    85 43       STA   $43
8008-    A9 4A       LDA   #$4A
800A-    85 3C       STA   $3C
800C-    A9 82       LDA   #$82
800E-    85 3D       STA   $3D
8010-    A9 6F       LDA   #$6F
8012-    85 3E       STA   $3E
8014-    A9 89       LDA   #$89
8016-    85 3F       STA   $3F
8018-    2C 89 C0    BIT   $C089
801B-    2C 89 C0    BIT   $C089
801E-    A0 00       LDY   #$00
8020-    20 2C FE    JSR   $FE2C
8023-    2C 81 C0    BIT   $C081
8026-    2C 81 C0    BIT   $C081
8029-    A9 EA       LDA   #$EA
802B-    8D 82 FD    STA   $FD82
802E-    8D 83 FD    STA   $FD83
8031-    A9 4C       LDA   #$4C
8033-    8D 7D E0    STA   $E07D
8036-    A9 D4       LDA   #$D4
8038-    8D 7E E0    STA   $E07E
803B-    A9 FC       LDA   #$FC
803D-    8D 7F E0    STA   $E07F
8040-    A9 20       LDA   #$20
8042-    8D 6D D5    STA   $D56D
8045-    8D A8 D5    STA   $D5A8
8048-    A9 25       LDA   #$25
804A-    8D 6E D5    STA   $D56E
804D-    8D A9 D5    STA   $D5A9
8050-    A9 FD       LDA   #$FD
8052-    8D 6F D5    STA   $D56F
8055-    8D AA D5    STA   $D5AA
8058-    A9 6E       LDA   #$6E
805A-    85 3C       STA   $3C
805C-    A9 81       LDA   #$81
805E-    85 3D       STA   $3D
8060-    A9 AF       LDA   #$AF
8062-    85 3E       STA   $3E
8064-    A9 D5       LDA   #$D5
8066-    85 3F       STA   $3F
```

COMPUKEYSOFT

```
8068-   A0 09      LDY  #$09
806A-   20 38 81   JSR  $8138
806D-   A9 41      LDA  #$41
806F-   85 3C      STA  $3C
8071-   A9 81      LDA  #$81
8073-   85 3D      STA  $3D
8075-   A9 B0      LDA  #$B0
8077-   85 3E      STA  $3E
8079-   A9 D8      LDA  #$D8
807B-   85 3F      STA  $3F
807D-   A0 19      LDY  #$19
807F-   20 38 81   JSR  $8138
8082-   A9 5A      LDA  #$5A
8084-   85 3C      STA  $3C
8086-   A9 81      LDA  #$81
8088-   85 3D      STA  $3D
808A-   A9 1B      LDA  #$1B
808C-   85 3E      STA  $3E
808E-   A9 FD      LDA  #$FD
8090-   85 3F      STA  $3F
8092-   A0 14      LDY  #$14
8094-   20 38 81   JSR  $8138
8097-   A9 77      LDA  #$77
8099-   85 3C      STA  $3C
809B-   A9 81      LDA  #$81
809D-   85 3D      STA  $3D
809F-   A9 C9      LDA  #$C9
80A1-   85 3E      STA  $3E
80A3-   A9 D8      LDA  #$D8
80A5-   85 3F      STA  $3F
80A7-   A0 27      LDY  #$27
80A9-   20 38 81   JSR  $8138
80AC-   A9 9E      LDA  #$9E
80AE-   85 3C      STA  $3C
80B0-   A9 81      LDA  #$81
80B2-   85 3D      STA  $3D
80B4-   A9 C9      LDA  #$C9
80B6-   85 3E      STA  $3E
80B8-   A9 FC      LDA  #$FC
80BA-   85 3F      STA  $3F
80BC-   A0 43      LDY  #$43
80BE-   20 38 81   JSR  $8138
80C1-   A9 E1      LDA  #$E1
80C3-   85 3C      STA  $3C
80C5-   A9 81      LDA  #$81
80C7-   85 3D      STA  $3D
80C9-   A9 FD      LDA  #$FD
80CB-   85 3E      STA  $3E
80CD-   A9 FE      LDA  #$FE
80CF-   85 3F      STA  $3F
80D1-   A0 2E      LDY  #$2E
80D3-   20 38 81   JSR  $8138
80D6-   A9 0F      LDA  #$0F
80D8-   85 3C      STA  $3C
80DA-   A9 82      LDA  #$82
80DC-   85 3D      STA  $3D
80DE-   A9 9F      LDA  #$9F
80E0-   85 3E      STA  $3E
80E2-   A9 F3      LDA  #$F3
80E4-   85 3F      STA  $3F
80E6-   A0 27      LDY  #$27
80E8-   20 38 81   JSR  $8138
80EB-   A9 36      LDA  #$36
80ED-   85 3C      STA  $3C
80EF-   A9 82      LDA  #$82
80F1-   85 3D      STA  $3D
80F3-   A9 75      LDA  #$75
80F5-   85 3E      STA  $3E
80F7-   A9 F7      LDA  #$F7
80F9-   85 3F      STA  $3F
80FB-   A0 14      LDY  #$14
80FD-   20 38 81   JSR  $8138
8100-   A9 4C      LDA  #$4C
8102-   8D AD FF   STA  $FFAD
8105-   A9 FD      LDA  #$FD
8107-   8D AE FF   STA  $FFAE
810A-   A9 FE      LDA  #$FE
810C-   8D AF FF   STA  $FFAF
```

```
810F-   A9 02      LDA  #$02
8111-   8D FC FF   STA  $FFFC
8114-   A9 FD      LDA  #$FD
8116-   8D FD FF   STA  $FFFD
8119-   A9 39      LDA  #$39
811B-   8D DC FF   STA  $FFDC
811E-   8D DE FF   STA  $FFDE
8121-   A9 57      LDA  #$57
8123-   8D 50 D0   STA  $D050
8126-   8D 4E D0   STA  $D04E
8129-   8D 34 D0   STA  $D034
812C-   A9 FF      LDA  #$FF
812E-   8D 51 D0   STA  $D051
8131-   8D 4F D0   STA  $D04F
8134-   8D 35 D0   STA  $D035
8137-   60         RTS
8138-   88         DEY
8139-   B1 3C      LDA  ($3C),Y
813B-   91 3E      STA  ($3E),Y
813D-   88         DEY
813E-   10 F9      BPL  $8139
8140-   60         RTS
8141-   4C CA FC   JMP  $FCCA
8144-   EA         NOP
8145-   EA         NOP
8146-   EA         NOP
8147-   EA         NOP
8148-   EA         NOP
8149-   EA         NOP
814A-   EA         NOP
814B-   EA         NOP
814C-   EA         NOP
814D-   EA         NOP
814E-   EA         NOP
814F-   EA         NOP
8150-   EA         NOP
8151-   EA         NOP
8152-   EA         NOP
8153-   EA         NOP
8154-   EA         NOP
8155-   EA         NOP
8156-   EA         NOP
8157-   EA         NOP
8158-   EA         NOP
8159-   EA         NOP
815A-   2C 88 C0   BIT  $C088
815D-   20 00 D0   JSR  $D000
8160-   2C 80 C0   BIT  $C080
8163-   60         RTS
8164-   2C 88 C0   BIT  $C088
8167-   20 06 D2   JSR  $D206
816A-   2C 80 C0   BIT  $C080
816D-   60         RTS
816E-   85 3C      STA  $3C
8170-   B1 9D      LDA  ($9D),Y
8172-   85 3D      STA  $3D
8174-   4C DE FC   JMP  $FCDE
8177-   4C F6 FC   JMP  $FCF6
817A-   EA         NOP
817B-   EA         NOP
817C-   EA         NOP
817D-   EA         NOP
817E-   EA         NOP
817F-   EA         NOP
8180-   EA         NOP
8181-   EA         NOP
8182-   EA         NOP
8183-   EA         NOP
8184-   EA         NOP
8185-   EA         NOP
8186-   EA         NOP
8187-   EA         NOP
8188-   EA         NOP
8189-   EA         NOP
818A-   EA         NOP
818B-   EA         NOP
818C-   EA         NOP
818D-   EA         NOP
```

```
818E-   EA         NOP
818F-   EA         NOP
8190-   EA         NOP
8191-   EA         NOP
8192-   EA         NOP
8193-   EA         NOP
8194-   EA         NOP
8195-   EA         NOP
8196-   EA         NOP
8197-   EA         NOP
8198-   EA         NOP
8199-   EA         NOP
819A-   EA         NOP
819B-   EA         NOP
819C-   EA         NOP
819D-   EA         NOP
819E-   60         RTS
819F-   2C 88 C0   BIT  $C088
81A2-   20 28 D2   JSR  $D228
81A5-   2C 80 C0   BIT  $C080
81A8-   60         RTS
81A9-   2C 88 C0   BIT  $C088
81AC-   20 14 D2   JSR  $D214
81AF-   2C 80 C0   BIT  $C080
81B2-   60         RTS
81B3-   2C 88 C0   BIT  $C088
81B6-   4C 29 D2   JMP  $D229
81B9-   2C 80 C0   BIT  $C080
81BC-   4C A2 D5   JMP  $D5A2
81BF-   2C 80 C0   BIT  $C080
81C2-   4C B8 D5   JMP  $D5B8
81C5-   2C 80 C0   BIT  $C080
81C8-   4C F9 D5   JMP  $D5F9
81CB-   2C 88 C0   BIT  $C088
81CE-   4C 5C D2   JMP  $D25C
81D1-   2C 80 C0   BIT  $C080
81D4-   4C 3C D4   JMP  $D43C
81D7-   2C 80 C0   BIT  $C080
81DA-   2C 80 C0   BIT  $C080
81DD-   EA         NOP
81DE-   4C 62 FA   JMP  $FA62
81E1-   B9 00 02   LDA  $0200,Y
81E4-   C9 E1      CMP  #$E1
81E6-   90 06      BCC  $81EE
81E8-   C9 FB      CMP  #$FB
81EA-   B0 02      BCS  $81EE
81EC-   29 DF      AND  #$DF
81EE-   4C B0 FF   JMP  $FFB0
81F1-   2C 80 C0   BIT  $C080
81F4-   20 1A D6   JSR  $D61A
81F7-   2C 88 C0   BIT  $C088
81FA-   60         RTS
81FB-   2C 80 C0   BIT  $C080
81FE-   20 0C DA   JSR  $DA0C
8201-   2C 88 C0   BIT  $C088
8204-   60         RTS
8205-   2C 88 C0   BIT  $C088
8208-   20 D7 D3   JSR  $D3D7
820B-   2C 80 C0   BIT  $C080
820E-   60         RTS
820F-   2C 83 C0   BIT  $C083
8212-   2C 83 C0   BIT  $C083
8215-   A9 60      LDA  #$60
8217-   8D 0C D5   STA  $D50C
821A-   A9 EA      LDA  #$EA
821C-   8D 8C D6   STA  $D68C
821F-   20 44 D4   JSR  $D444
8222-   2C 83 C0   BIT  $C083
8225-   2C 83 C0   BIT  $C083
8228-   A9 4C      LDA  #$4C
822A-   8D 0C D5   STA  $D50C
822D-   A9 9A      LDA  #$9A
822F-   8D 8C D6   STA  $D68C
8232-   2C 88 C0   BIT  $C088
8235-   60         RTS
8236-   2C 80 C0   BIT  $C080
8239-   20 24 ED   JSR  $ED24
823C-   2C 88 C0   BIT  $C088
```

COMPUKEYSOFT

```
823F-   60          RTS
8240-   2C 80 C0    BIT   $C080
8243-   20 2C D7    JSR   $D72C
8246-   8D 88 C0    STA   $C088
8249-   60          RTS
824A-   86 46       STX   $46
824C-   91 28       STA   ($28),Y
824E-   2C 8B C0    BIT   $C08B
8251-   2C 8B C0    BIT   $C08B
8254-   A9 20       LDA   #$20
8256-   8D 5B D2    STA   $D25B
8259-   2C 88 C0    BIT   $C088
825C-   20 52 D1    JSR   $D152
825F-   C9 9B       CMP   #$9B
8261-   F0 11       BEQ   $8274
8263-   2C 8B C0    BIT   $C08B
8266-   2C 8B C0    BIT   $C08B
8269-   A9 20       LDA   #$20
826B-   8D 5B D2    STA   $D25B
826E-   2C 88 C0    BIT   $C088
8271-   4C F0 D0    JMP   $D0F0
8274-   2C 8B C0    BIT   $C08B
8277-   2C 8B C0    BIT   $C08B
827A-   A9 2B       LDA   #$2B
827C-   8D 5B D2    STA   $D25B
827F-   2C 88 C0    BIT   $C088
8282-   2C 10 C0    BIT   $C010
8285-   20 52 D1    JSR   $D152
8288-   2C 10 C0    BIT   $C010
828B-   48          PHA
828C-   20 CB D1    JSR   $D1CB
828F-   68          PLA
8290-   C9 D4       CMP   #$D4
8292-   D0 14       BNE   $82A8
8294-   A9 40       LDA   #$40
8296-   2C 8B C0    BIT   $C08B
8299-   2C 8B C0    BIT   $C08B
829C-   4D 59 D2    EOR   $D259
829F-   8D 59 D2    STA   $D259
82A2-   2C 88 C0    BIT   $C088
82A5-   4C 04 D0    JMP   $D004
82A8-   C9 CE       CMP   #$CE
82AA-   D0 06       BNE   $82B2
82AC-   20 52 D1    JSR   $D152
82AF-   4C 4C D1    JMP   $D14C
82B2-   C9 CC       CMP   #$CC
82B4-   D0 17       BNE   $82CD
82B6-   A9 7F       LDA   #$7F
82B8-   2D 59 D2    AND   $D259
82BB-   2C 8B C0    BIT   $C08B
82BE-   2C 8B C0    BIT   $C08B
82C1-   8D 59 D2    STA   $D259
82C4-   2C 88 C0    BIT   $C088
82C7-   2C 10 C0    BIT   $C010
82CA-   4C 04 D0    JMP   $D004
82CD-   C9 D5       CMP   #$D5
82CF-   D0 17       BNE   $82E8
82D1-   A9 80       LDA   #$80
82D3-   0D 59 D2    ORA   $D259
82D6-   2C 8B C0    BIT   $C08B
82D9-   2C 8B C0    BIT   $C08B
82DC-   8D 59 D2    STA   $D259
82DF-   2C 88 C0    BIT   $C088
82E2-   2C 10 C0    BIT   $C010
82E5-   4C 04 D0    JMP   $D004
82E8-   C9 AC       CMP   #$AC
82EA-   D0 05       BNE   $82F1
82EC-   A9 9C       LDA   #$9C
82EE-   4C 4C D1    JMP   $D14C
82F1-   C9 AF       CMP   #$AF
82F3-   D0 05       BNE   $82FA
82F5-   A9 9F       LDA   #$9F
82F7-   4C 4C D1    JMP   $D14C
82FA-   C9 A7       CMP   #$A7
82FC-   D0 05       BNE   $8303
82FE-   A9 E0       LDA   #$E0
8300-   4C 4C D1    JMP   $D14C
8303-   C9 B0       CMP   #$B0
8305-   90 0D       BCC   $8314
8307-   C9 BA       CMP   #$BA
8309-   B0 09       BCS   $8314
830B-   29 0F       AND   #$0F
830D-   AA          TAX
830E-   BD 4F D2    LDA   $D24F,X
8311-   4C 4C D1    JMP   $D14C
8314-   2C 5A D2    BIT   $D25A
8317-   50 03       BVC   $831C
8319-   4C 04 D0    JMP   $D004
831C-   C9 CF       CMP   #$CF
831E-   B0 13       BCS   $8333
8320-   C9 C9       CMP   #$C9
8322-   90 0F       BCC   $8333
8324-   C9 CC       CMP   #$CC
8326-   F0 0B       BEQ   $8333
8328-   A8          TAY
8329-   B9 48 FA    LDA   $FA48,Y
832C-   38          SEC
832D-   20 2C FC    JSR   $FC2C
8330-   4C 2A D0    JMP   $D02A
8333-   38          SEC
8334-   20 2C FC    JSR   $FC2C
8337-   4C 04 D0    JMP   $D004
833A-   AD 00 C0    LDA   $C000
833D-   C9 95       CMP   #$95
833F-   D0 0A       BNE   $834B
8341-   2C 5A D2    BIT   $D25A
8344-   70 02       BVS   $8348
8346-   B1 28       LDA   ($28),Y
8348-   4C 4C D1    JMP   $D14C
834B-   AD 00 C0    LDA   $C000
834E-   2C 59 D2    BIT   $D259
8351-   30 43       BMI   $8396
8353-   C9 A0       CMP   #$A0
8355-   90 33       BCC   $838A
8357-   AD 63 C0    LDA   $C063
835A-   4D 59 D2    EOR   $D259
835D-   29 80       AND   #$80
835F-   4A          LSR
8360-   4A          LSR
8361-   0D 00 C0    ORA   $C000
8364-   C9 FD       CMP   #$FD
8366-   D0 02       BNE   $836A
8368-   A9 ED       LDA   #$ED
836A-   C9 FE       CMP   #$FE
836C-   D0 02       BNE   $8370
836E-   A9 EE       LDA   #$EE
8370-   C9 E0       CMP   #$E0
8372-   D0 02       BNE   $8376
8374-   A9 F0       LDA   #$F0
8376-   C9 DD       CMP   #$DD
8378-   D0 02       BNE   $837C
837A-   A9 CD       LDA   #$CD
837C-   C9 DE       CMP   #$DE
837E-   D0 02       BNE   $8382
8380-   A9 CE       LDA   #$CE
8382-   C9 C0       CMP   #$C0
8384-   D0 10       BNE   $8396
8386-   A9 D0       LDA   #$D0
8388-   D0 0C       BNE   $8396
838A-   C9 90       CMP   #$90
838C-   D0 02       BNE   $8390
838E-   A9 C0       LDA   #$C0
8390-   C9 8E       CMP   #$8E
8392-   D0 02       BNE   $8396
8394-   A9 DE       LDA   #$DE
8396-   A6 46       LDX   $46
8398-   2C 10 C0    BIT   $C010
839B-   60          RTS
839C-   20 DE D1    JSR   $D1DE
839F-   2C 5A D2    BIT   $D25A
83A2-   50 3A       BVC   $83DE
83A4-   AD 56 AA    LDA   $AA56
83A7-   C9 C3       CMP   #$C3
83A9-   D0 33       BNE   $83DE
83AB-   A4 3E       LDY   $3E
83AD-   AD 5B D2    LDA   $D25B
83B0-   C9 2B       CMP   #$2B
83B2-   F0 03       BEQ   $83B7
83B4-   B9 00 02    LDA   $0200,Y
83B7-   09 80       ORA   #$80
83B9-   C9 80       CMP   #$80
83BB-   D0 02       BNE   $83BF
83BD-   A9 A0       LDA   #$A0
83BF-   C9 A0       CMP   #$A0
83C1-   B0 02       BCS   $83C5
83C3-   69 40       ADC   #$40
83C5-   48          PHA
83C6-   A9 3F       LDA   #$3F
83C8-   85 32       STA   $32
83CA-   68          PLA
83CB-   20 ED FD    JSR   $FDED
83CE-   A9 FF       LDA   #$FF
83D0-   85 32       STA   $32
83D2-   A9 88       LDA   #$88
83D4-   20 ED FD    JSR   $FDED
83D7-   2C 00 C0    BIT   $C000
83DA-   10 FB       BPL   $83D7
83DC-   30 24       BMI   $8402
83DE-   A4 24       LDY   $24
83E0-   B1 28       LDA   ($28),Y
83E2-   AA          TAX
83E3-   AD 5B D2    LDA   $D25B
83E6-   91 28       STA   ($28),Y
83E8-   2C 00 C0    BIT   $C000
83EB-   30 05       BMI   $83F2
83ED-   A9 31       LDA   #$31
83EF-   20 BC D1    JSR   $D1BC
83F2-   8A          TXA
83F3-   91 28       STA   ($28),Y
83F5-   2C 00 C0    BIT   $C000
83F8-   30 08       BMI   $8402
83FA-   A9 44       LDA   #$44
83FC-   20 BC D1    JSR   $D1BC
83FF-   4C 94 D1    JMP   $D194
8402-   AD 00 C0    LDA   $C000
8405-   60          RTS
8406-   48          PHA
8407-   AD 00 C0    LDA   $C000
840A-   30 07       BMI   $8413
840C-   68          PLA
840D-   38          SEC
840E-   E9 01       SBC   #$01
8410-   D0 F4       BNE   $8406
8412-   48          PHA
8413-   68          PLA
8414-   60          RTS
8415-   2C 59 D2    BIT   $D259
8418-   50 0D       BVC   $8427
841A-   A0 06       LDY   #$06
841C-   A9 0A       LDA   #$0A
841E-   20 BC D1    JSR   $D1BC
8421-   AD 30 C0    LDA   $C030
8424-   88          DEY
8425-   D0 F5       BNE   $841C
8427-   60          RTS
8428-   A6 4E       LDX   $4E
842A-   A4 4F       LDY   $4F
842C-   06 4E       ASL   $4E
842E-   26 4F       ROL   $4F
8430-   06 4E       ASL   $4E
8432-   26 4F       ROL   $4F
8434-   8A          TXA
8435-   45 4E       EOR   $4E
8437-   AA          TAX
8438-   98          TYA
8439-   45 4F       EOR   $4F
843B-   A8          TAY
843C-   06 4E       ASL   $4E
843E-   66 4F       ROR   $4F
8440-   8A          TXA
8441-   45 4E       EOR   $4E
8443-   18          CLC
8444-   69 05       ADC   #$05
8446-   85 4E       STA   $4E
```

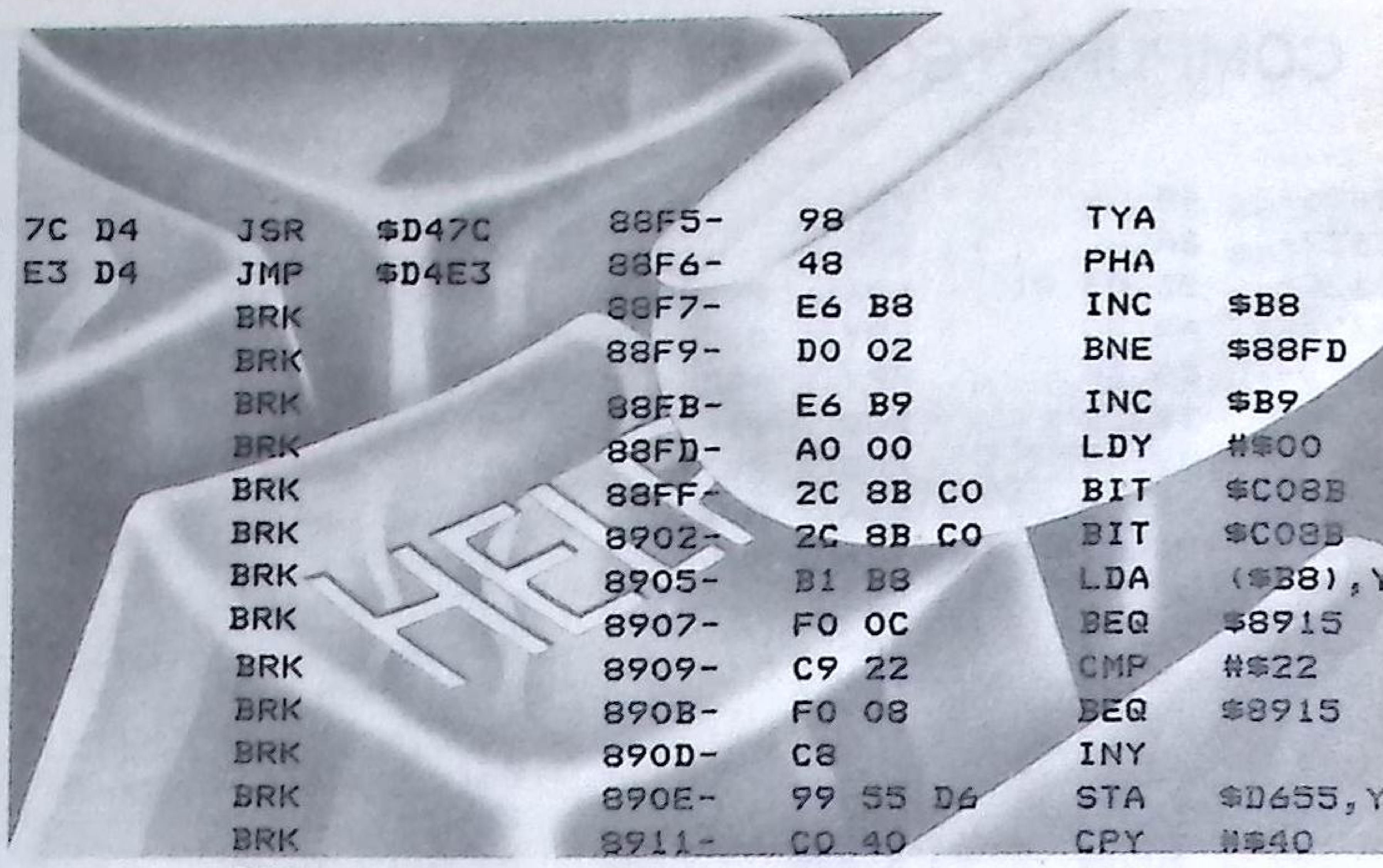
COMPUKEYSOFT

```
8448-   98          TYA
8449-   45 4F       EOR   $4F
844B-   69 00       ADC   #$00
844D-   85 4F       STA   $4F
844F-   60          RTS
8450-   BD 00 02    LDA   $0200,X
8453-   C9 61       CMP   #$61
8455-   90 06       BCC   $845D
8457-   C9 7B       CMP   #$7B
8459-   B0 02       BCS   $845D
845B-   29 DF       AND   #$DF
845D-   60          RTS
845E-   C9 41       CMP   #$41
8460-   90 0D       BCC   $846F
8462-   C9 5B       CMP   #$5B
8464-   90 0A       BCC   $8470
8466-   C9 61       CMP   #$61
8468-   90 05       BCC   $846F
846A-   C9 7B       CMP   #$7B
846C-   90 02       BCC   $8470
846E-   18          CLC
846F-   60          RTS
8470-   38          SEC
8471-   60          RTS
8472-   60          RTS
8473-   38          SEC
8474-   A5 3D       LDA   $3D
8476-   E5 3C       SBC   $3C
8478-   D0 03       BNE   $847D
847A-   4C E4 FC    JMP   $FCE4
847D-   C9 80       CMP   #$80
847F-   D0 03       BNE   $8484
8481-   4C EA FC    JMP   $FCEA
8484-   A5 3C       LDA   $3C
8486-   20 14 D2    JSR   $D214
8489-   90 0B       BCC   $8496
848B-   09 20       ORA   #$20
848D-   38          SEC
848E-   E5 3D       SBC   $3D
8490-   F0 E8       BEQ   $847A
8492-   C9 80       CMP   #$80
8494-   F0 EB       BEQ   $8481
8496-   4C F0 FC    JMP   $FCF0
8499-   FF          ???
849A-   DB          ???
849B-   DC          ???
849C-   DD DE DF    CMP   $DFDE,X
849F-   FB          ???
84A0-   FC          ???
84A1-   FD FE C0    SBC   $C0FE,X
84A4-   00          BRK
84A5-   20 08 48    JSR   $4808
84A8-   2C 8B C0    BIT   $C08B
84AB-   2C 8B C0    BIT   $C08B
84AE-   A9 00       LDA   #$00
84B0-   8D 55 D6    STA   $D655
84B3-   2C 88 C0    BIT   $C088
84B6-   A5 76       LDA   $76
84B8-   C9 FF       CMP   #$FF
84BA-   F0 05       BEQ   $84C1
84BC-   68          PLA
84BD-   28          PLP
84BE-   4C 27 FF    JMP   $FF27
84C1-   68          PLA
84C2-   28          PLP
84C3-   90 17       BCC   $84DC
84C5-   F0 15       BEQ   $84DC
84C7-   C9 C9       CMP   #$C9
84C9-   F0 11       BEQ   $84DC
84CB-   C9 2C       CMP   #$2C
84CD-   F0 0D       BEQ   $84DC
84CF-   C9 22       CMP   #$22
84D1-   D0 06       BNE   $84D9
84D3-   20 AB D6    JSR   $D6AB
84D6-   4C 92 D2    JMP   $D292
84D9-   4C 27 FF    JMP   $FF27
84DC-   20 17 FF    JSR   $FF17
84DF-   20 0D FF    JSR   $FF0D
84E2-   20 B7 00    JSR   $00B7
84E5-   F0 30       BEQ   $8517
84E7-   C9 C9       CMP   #$C9
84E9-   F0 0E       BEQ   $84F9
84EB-   C9 2C       CMP   #$2C
84ED-   F0 0A       BEQ   $84F9
84EF-   C9 22       CMP   #$22
84F1-   D0 E6       BNE   $84D9
84F3-   20 AB D6    JSR   $D6AB
84F6-   4C CD D2    JMP   $D2CD
84F9-   20 B1 00    JSR   $00B1
84FC-   20 17 FF    JSR   $FF17
84FF-   20 B7 00    JSR   $00B7
8502-   C9 22       CMP   #$22
8504-   D0 06       BNE   $850C
8506-   20 AB D6    JSR   $D6AB
8509-   4C CD D2    JMP   $D2CD
850C-   C9 3A       CMP   #$3A
850E-   F0 07       BEQ   $8517
8510-   C9 00       CMP   #$00
8512-   F0 03       BEQ   $8517
8514-   4C 8F D2    JMP   $D28F
8517-   68          PLA
8518-   68          PLA
8519-   20 33 FB    JSR   $FB33
851C-   A5 50       LDA   $50
851E-   05 51       ORA   $51
8520-   D0 06       BNE   $8528
8522-   A9 FF       LDA   #$FF
8524-   85 50       STA   $50
8526-   85 51       STA   $51
8528-   A5 50       LDA   $50
852A-   85 E4       STA   $E4
852C-   A5 51       LDA   $51
852E-   85 E5       STA   $E5
8530-   20 2F FB    JSR   $FB2F
8533-   A9 8C       LDA   #$8C
8535-   20 ED FD    JSR   $FDED
8538-   20 58 FC    JSR   $FC58
853B-   A0 01       LDY   #$01
853D-   B1 9B       LDA   ($9B),Y
853F-   D0 03       BNE   $8544
8541-   4C 90 D3    JMP   $D390
8544-   C8          INY
8545-   B1 9B       LDA   ($9B),Y
8547-   AA          TAX
8548-   C8          INY
8549-   B1 9B       LDA   ($9B),Y
854B-   C5 51       CMP   $51
854D-   D0 04       BNE   $8553
854F-   E4 50       CPX   $50
8551-   F0 05       BEQ   $8558
8553-   90 03       BCC   $8558
8555-   4C 90 D3    JMP   $D390
8558-   84 85       STY   $85
855A-   A5 36       LDA   $36
855C-   48          PHA
855D-   A5 37       LDA   $37
855F-   48          PHA
8560-   A9 21       LDA   #$21
8562-   85 36       STA   $36
8564-   A9 FF       LDA   #$FF
8566-   85 37       STA   $37
8568-   A9 00       LDA   #$00
856A-   85 30       STA   $30
856C-   A4 85       LDY   $85
856E-   B1 9B       LDA   ($9B),Y
8570-   20 75 F7    JSR   $F775
8573-   68          PLA
8574-   85 37       STA   $37
8576-   68          PLA
8577-   85 36       STA   $36
8579-   A9 20       LDA   #$20
857B-   A4 85       LDY   $85
857D-   29 7F       AND   #$7F
857F-   20 D7 D3    JSR   $D3D7
8582-   C8          INY
8583-   B1 9B       LDA   ($9B),Y
8585-   D0 6C       BNE   $85F3
8587-   20 D7 D3    JSR   $D3D7
858A-   A0 02       LDY   #$02
858C-   B1 9B       LDA   ($9B),Y
858E-   AA          TAX
858F-   48          PHA
8590-   C8          INY
8591-   B1 9B       LDA   ($9B),Y
8593-   48          PHA
8594-   AD 55 D6    LDA   $D655
8597-   F0 0F       BEQ   $85A8
8599-   A0 00       LDY   #$00
859B-   20 7C D4    JSR   $D47C
859E-   68          PLA
859F-   48          PHA
85A0-   20 75 F7    JSR   $F775
85A3-   20 DE D6    JSR   $D6DE
85A6-   90 0B       BCC   $85B3
85A8-   20 B6 D4    JSR   $D4B6
85AB-   A9 8C       LDA   #$8C
85AD-   20 ED FD    JSR   $FDED
85B0-   20 58 FC    JSR   $FC58
85B3-   68          PLA
85B4-   85 51       STA   $51
85B6-   68          PLA
85B7-   85 50       STA   $50
85B9-   20 0D FF    JSR   $FF0D
85BC-   A5 E4       LDA   $E4
85BE-   85 50       STA   $50
85C0-   A5 E5       LDA   $E5
85C2-   85 51       STA   $51
85C4-   AD 5A D2    LDA   $D25A
85C7-   D0 11       BNE   $85DA
85C9-   A0 00       LDY   #$00
85CB-   B1 9B       LDA   ($9B),Y
85CD-   AA          TAX
85CE-   C8          INY
85CF-   B1 9B       LDA   ($9B),Y
85D1-   86 9B       STX   $9B
85D3-   85 9C       STA   $9C
85D5-   F0 03       BEQ   $85DA
85D7-   4C F1 D2    JMP   $D2F1
85DA-   A9 0D       LDA   #$0D
85DC-   20 D7 D3    JSR   $D3D7
85DF-   A9 00       LDA   #$00
85E1-   2C 8B C0    BIT   $C08B
85E4-   2C 8B C0    BIT   $C08B
85E7-   8D 5A D2    STA   $D25A
85EA-   8D 55 D6    STA   $D655
85ED-   2C 88 C0    BIT   $C088
85F0-   4C FC FC    JMP   $FCFC
85F3-   10 8A       BPL   $857F
85F5-   38          SEC
85F6-   E9 7F       SBC   #$7F
85F8-   AA          TAX
85F9-   84 85       STY   $85
85FB-   A0 D0       LDY   #$D0
85FD-   84 9D       STY   $9D
85FF-   A0 CF       LDY   #$CF
8601-   84 9E       STY   $9E
8603-   A0 FF       LDY   #$FF
8605-   CA          DEX
8606-   F0 07       BEQ   $860F
8608-   20 7F F7    JSR   $F77F
860B-   10 FB       BPL   $8608
860D-   30 F6       BMI   $8605
860F-   20 7F F7    JSR   $F77F
8612-   30 05       BMI   $8619
8614-   20 D7 D3    JSR   $D3D7
8617-   D0 F6       BNE   $860F
8619-   20 D7 D3    JSR   $D3D7
861C-   A4 85       LDY   $85
861E-   4C 38 D3    JMP   $D338
8621-   08          PHP
8622-   48          PHA
8623-   98          TYA
8624-   48          PHA
8625-   8A          TXA
```

COMPUKEYSOFT

```
8626-   48          PHA
8627-   BA          TSX
8628-   BD 03 01    LDA   $0103,X
862B-   A4 30       LDY   $30
862D-   C0 EF       CPY   #$EF
862F-   B0 07       BCS   $8638
8631-   29 7F       AND   #$7F
8633-   99 00 02    STA   $0200,Y
8636-   E6 30       INC   $30
8638-   68          PLA
8639-   AA          TAX
863A-   68          PLA
863B-   A8          TAY
863C-   68          PLA
863D-   28          PLP
863E-   60          RTS
863F-   98          TYA
8640-   48          PHA
8641-   A5 24       LDA   $24
8643-   48          PHA
8644-   A5 25       LDA   $25
8646-   48          PHA
8647-   A9 00       LDA   #$00
8649-   85 24       STA   $24
864B-   85 25       STA   $25
864D-   20 22 FC    JSR   $FC22
8650-   AD 56 AA    LDA   $AA56
8653-   C9 C3       CMP   #$C3
8655-   D0 03       BNE   $865A
8657-   20 56 D4    JSR   $D456
865A-   A0 00       LDY   #$00
865C-   B9 00 02    LDA   $0200,Y
865F-   09 80       ORA   #$80
8661-   C9 A0       CMP   #$A0
8663-   B0 0D       BCS   $8672
8665-   C9 80       CMP   #$80
8667-   F0 17       BEQ   $8680
8669-   48          PHA
866A-   A9 3F       LDA   #$3F
866C-   85 32       STA   $32
866E-   68          PLA
866F-   18          CLC
8670-   69 40       ADC   #$40
8672-   20 ED FD    JSR   $FDED
8675-   A9 FF       LDA   #$FF
8677-   85 32       STA   $32
8679-   C8          INY
867A-   C0 F0       CPY   #$F0
867C-   90 DE       BCC   $865C
867E-   B0 0A       BCS   $868A
8680-   A9 A0       LDA   #$A0
8682-   20 ED FD    JSR   $FDED
8685-   C8          INY
8686-   C0 F0       CPY   #$F0
8688-   90 F6       BCC   $8680
868A-   68          PLA
868B-   85 25       STA   $25
868D-   68          PLA
868E-   85 24       STA   $24
8690-   20 22 FC    JSR   $FC22
8693-   AD 56 AA    LDA   $AA56
8696-   C9 C3       CMP   #$C3
8698-   D0 03       BNE   $869D
869A-   20 56 D4    JSR   $D456
869D-   68          PLA
869E-   A8          TAY
869F-   60          RTS
86A0-   48          PHA
86A1-   A5 25       LDA   $25
86A3-   48          PHA
86A4-   A5 24       LDA   $24
86A6-   48          PHA
86A7-   A9 00       LDA   #$00
86A9-   85 24       STA   $24
86AB-   85 25       STA   $25
86AD-   A9 9E       LDA   #$9E
86AF-   20 ED FD    JSR   $FDED
86B2-   68          PLA
86B3-   09 80       ORA   #$80
86B5-   18          CLC
86B6-   69 20       ADC   #$20
86B8-   20 ED FD    JSR   $FDED
86BB-   68          PLA
86BC-   09 80       ORA   #$80
86BE-   18          CLC
86BF-   69 20       ADC   #$20
86C1-   20 ED FD    JSR   $FDED
86C4-   68          PLA
86C5-   60          RTS
86C6-   A9 00       LDA   #$00
86C8-   85 25       STA   $25
86CA-   85 24       STA   $24
86CC-   C0 F0       CPY   #$F0
86CE-   90 02       BCC   $86D2
86D0-   A0 EF       LDY   #$EF
86D2-   AD 56 AA    LDA   $AA56
86D5-   C9 C3       CMP   #$C3
86D7-   D0 04       BNE   $86DD
86D9-   A9 4F       LDA   #$4F
86DB-   D0 02       BNE   $86DF
86DD-   A9 27       LDA   #$27
86DF-   85 4E       STA   $4E
86E1-   98          TYA
86E2-   C5 4E       CMP   $4E
86E4-   F0 02       BEQ   $86E8
86E6-   B0 10       BCS   $86F8
86E8-   85 24       STA   $24
86EA-   20 22 FC    JSR   $FC22
86ED-   AD 56 AA    LDA   $AA56
86F0-   C9 C3       CMP   #$C3
86F2-   D0 03       BNE   $86F7
86F4-   20 56 D4    JSR   $D456
86F7-   60          RTS
86F8-   E6 25       INC   $25
86FA-   18          CLC
86FB-   E5 4E       SBC   $4E
86FD-   4C 98 D4    JMP   $D498
8700-   08          PHP
8701-   48          PHA
8702-   98          TYA
8703-   48          PHA
8704-   8A          TXA
8705-   48          PHA
8706-   A9 40       LDA   #$40
8708-   2C 8B C0    BIT   $C08B
870B-   2C 8B C0    BIT   $C08B
870E-   8D 5A D2    STA   $D25A
8711-   2C 88 C0    BIT   $C088
8714-   A4 30       LDY   $30
8716-   88          DEY
8717-   84 30       STY   $30
8719-   A0 00       LDY   #$00
871B-   84 4F       STY   $4F
871D-   20 7C D4    JSR   $D47C
8720-   20 F5 D3    JSR   $D3F5
8723-   AD 56 AA    LDA   $AA56
8726-   C9 C3       CMP   #$C3
8728-   D0 03       BNE   $872D
872A-   20 7C D4    JSR   $D47C
872D-   86 46       STX   $46
872F-   84 3E       STY   $3E
8731-   98          TYA
8732-   48          PHA
8733-   20 04 D0    JSR   $D004
8736-   AA          TAX
8737-   68          PLA
8738-   A8          TAY
8739-   8A          TXA
873A-   C9 95       CMP   #$95
873C-   D0 0B       BNE   $8749
873E-   B9 00 02    LDA   $0200,Y
8741-   09 80       ORA   #$80
8743-   C9 80       CMP   #$80
8745-   D0 02       BNE   $8749
8747-   A9 A0       LDA   #$A0
8749-   C9 89       CMP   #$89
874B-   D0 12       BNE   $875F
874D-   0D 5A D2    ORA   $D25A
8750-   2C 8B C0    BIT   $C08B
8753-   2C 8B C0    BIT   $C08B
8756-   8D 5A D2    STA   $D25A
8759-   2C 88 C0    BIT   $C088
875C-   4C D3 D4    JMP   $D4D3
875F-   C9 84       CMP   #$84
8761-   D0 03       BNE   $8766
8763-   4C EA D5    JMP   $D5EA
8766-   C9 88       CMP   #$88
8768-   D0 03       BNE   $876D
876A-   4C E0 D5    JMP   $D5E0
876D-   C9 8E       CMP   #$8E
876F-   D0 03       BNE   $8774
8771-   4C 00 D6    JMP   $D600
8774-   C9 97       CMP   #$97
8776-   D0 03       BNE   $877B
8778-   4C 1A D6    JMP   $D61A
877B-   C9 98       CMP   #$98
877D-   D0 08       BNE   $8787
877F-   A0 F0       LDY   #$F0
8781-   20 7C D4    JSR   $D47C
8784-   4C D9 D5    JMP   $D5D9
8787-   C9 9A       CMP   #$9A
8789-   D0 03       BNE   $878E
878B-   4C 29 D6    JMP   $D629
878E-   C9 81       CMP   #$81
8790-   D0 03       BNE   $8795
8792-   4C 3C D6    JMP   $D63C
8795-   C9 93       CMP   #$93
8797-   D0 03       BNE   $879C
8799-   4C 47 D6    JMP   $D647
879C-   C9 94       CMP   #$94
879E-   F0 41       BEQ   $87E1
87A0-   C9 8D       CMP   #$8D
87A2-   F0 44       BEQ   $87E8
87A4-   29 7F       AND   #$7F
87A6-   2C 5A D2    BIT   $D25A
87A9-   10 24       BPL   $87CF
87AB-   84 4E       STY   $4E
87AD-   A4 30       LDY   $30
87AF-   C0 F0       CPY   #$F0
87B1-   90 0C       BCC   $87BF
87B3-   98          TYA
87B4-   48          PHA
87B5-   20 DD FB    JSR   $FBDD
87B8-   68          PLA
87B9-   A8          TAY
87BA-   20 08 D6    JSR   $D608
87BD-   F0 1F       BEQ   $87DE
87BF-   48          PHA
87C0-   C8          INY
87C1-   84 30       STY   $30
87C3-   B9 FF 01    LDA   $01FF,Y
87C6-   99 00 02    STA   $0200,Y
87C9-   88          DEY
87CA-   C4 4E       CPY   $4E
87CC-   D0 F5       BNE   $87C3
87CE-   68          PLA
87CF-   99 00 02    STA   $0200,Y
87D2-   C8          INY
87D3-   C4 30       CPY   $30
87D5-   90 07       BCC   $87DE
87D7-   84 30       STY   $30
87D9-   A9 00       LDA   #$00
87DB-   99 00 02    STA   $0200,Y
87DE-   4C D3 D4    JMP   $D4D3
87E1-   A9 00       LDA   #$00
87E3-   99 00 02    STA   $0200,Y
87E6-   84 30       STY   $30
87E8-   20 F5 D3    JSR   $D3F5
87EB-   A0 F0       LDY   #$F0
87ED-   20 7C D4    JSR   $D47C
87F0-   A9 00       LDA   #$00
87F2-   2C 8B C0    BIT   $C08B
87F5-   2C 8B C0    BIT   $C08B
87F8-   8D 5A D2    STA   $D25A
```

COMPUKEYSOFT



```
87FB-   2C 88 C0   BIT   $C088
87FE-   A6 30      LDX   $30
8800-   F0 0B      BEQ   $880D
8802-   BD FF 01   LDA   $01FF,X
8805-   29 7F      AND   #$7F
8807-   9D FF 01   STA   $01FF,X
880A-   CA         DEX
880B-   D0 F5      BNE   $8802
880D-   A2 FF      LDX   #$FF
880F-   86 B8      STX   $B8
8811-   A0 01      LDY   #$01
8813-   84 B9      STY   $B9
8815-   20 B1 00   JSR   $00B1
8818-   B0 09      BCS   $8823
881A-   A9 00      LDA   #$00
881C-   A2 FF      LDX   #$FF
881E-   A0 01      LDY   #$01
8820-   20 9F F3   JSR   $F39F
8823-   68         PLA
8824-   AA         TAX
8825-   68         PLA
8826-   A8         TAY
8827-   68         PLA
8828-   28         PLP
8829-   60         RTS
882A-   20 08 D6   JSR   $D608
882D-   98         TYA
882E-   F0 01      BEQ   $8831
8830-   88         DEY
8831-   4C D3 D4   JMP   $D4D3
8834-   C4 30      CPY   $30
8836-   B0 0F      BCS   $8847
8838-   98         TYA
8839-   AA         TAX
883A-   BD 01 02   LDA   $0201,X
883D-   9D 00 02   STA   $0200,X
8840-   E8         INX
8841-   E4 30      CPX   $30
8843-   90 F5      BCC   $883A
8845-   C6 30      DEC   $30
8847-   4C D3 D4   JMP   $D4D3
884A-   86 46      STX   $46
884C-   20 04 D0   JSR   $D004
884F-   4C 5A D5   JMP   $D55A
8852-   A9 7F      LDA   #$7F
8854-   2D 5A D2   AND   $D25A
8857-   2C 8B C0   BIT   $C08B
885A-   2C 8B C0   BIT   $C08B
885D-   8D 5A D2   STA   $D25A
8860-   2C 88 C0   BIT   $C088
8863-   60         RTS
8864-   C0 28      CPY   #$28
8866-   90 31      BCC   $8899
8868-   20 96 D6   JSR   $D696
886B-   98         TYA
886C-   38         SEC
886D-   E9 28      SBC   #$28
886F-   A8         TAY
8870-   4C 4F D6   JMP   $D64F
8873-   C0 C8      CPY   #$C8
8875-   B0 22      BCS   $8899
8877-   20 96 D6   JSR   $D696
887A-   98         TYA
887B-   18         CLC
887C-   69 28      ADC   #$28
887E-   C5 30      CMP   $30
8880-   B0 17      BCS   $8899
8882-   A8         TAY
8883-   4C 4F D6   JMP   $D64F
8886-   C0 01      CPY   #$01
8888-   90 0F      BCC   $8899
888A-   20 96 D6   JSR   $D696
888D-   88         DEY
888E-   4C 4F D6   JMP   $D64F
8891-   C4 30      CPY   $30
8893-   B0 04      BCS   $8899
8895-   20 96 D6   JSR   $D696
8898-   C8         INY
```

```
8899-   20 7C D4   JSR   $D47C
889C-   4C E3 D4   JMP   $D4E3
889F-   00         BRK
88A0-   00         BRK
88A1-   00         BRK
88A2-   00         BRK
88A3-   00         BRK
88A4-   00         BRK
88A5-   00         BRK
88A6-   00         BRK
88A7-   00         BRK
88A8-   00         BRK
88A9-   00         BRK
88AA-   00         BRK
88AB-   00         BRK
88AC-   00         BRK
88AD-   00         BRK
88AE-   00         BRK
88AF-   00         BRK
88B0-   00         BRK
88B1-   00         BRK
88B2-   00         BRK
88B3-   00         BRK
88B4-   00         BRK
88B5-   00         BRK
88B6-   00         BRK
88B7-   00         BRK
88B8-   00         BRK
88B9-   00         BRK
88BA-   00         BRK
88BB-   00         BRK
88BC-   00         BRK
88BD-   00         BRK
88BE-   00         BRK
88BF-   00         BRK
88C0-   00         BRK
88C1-   00         BRK
88C2-   00         BRK
88C3-   00         BRK
88C4-   00         BRK
88C5-   00         BRK
88C6-   00         BRK
88C7-   00         BRK
88C8-   00         BRK
88C9-   00         BRK
88CA-   00         BRK
88CB-   00         BRK
88CC-   00         BRK
88CD-   00         BRK
88CE-   00         BRK
88CF-   00         BRK
88D0-   00         BRK
88D1-   00         BRK
88D2-   00         BRK
88D3-   00         BRK
88D4-   00         BRK
88D5-   00         BRK
88D6-   00         BRK
88D7-   00         BRK
88D8-   00         BRK
88D9-   00         BRK
88DA-   00         BRK
88DB-   00         BRK
88DC-   00         BRK
88DD-   00         BRK
88DE-   00         BRK
88DF-   00         BRK
88E0-   48         PHA
88E1-   B9 00 02   LDA   $0200,Y
88E4-   09 80      ORA   #$80
88E6-   C9 80      CMP   #$80
88E8-   D0 02      BNE   $88EC
88EA-   A9 A0      LDA   #$A0
88EC-   C9 A0      CMP   #$A0
88EE-   90 03      BCC   $88F3
88F0-   20 ED FD   JSR   $FDED
88F3-   68         PLA
88F4-   60         RTS
```

```
88F5-   98         TYA
88F6-   48         PHA
88F7-   E6 B8      INC   $B8
88F9-   D0 02      BNE   $88FD
88FB-   E6 B9      INC   $B9
88FD-   A0 00      LDY   #$00
88FF-   2C 8B C0   BIT   $C08B
8902-   2C 8B C0   BIT   $C08B
8905-   B1 B8      LDA   ($B8),Y
8907-   F0 0C      BEQ   $8915
8909-   C9 22      CMP   #$22
890B-   F0 08      BEQ   $8915
890D-   C8         INY
890E-   99 55 D6   STA   $D655,Y
8911-   C0 40      CPY   #$40
8913-   90 F0      BCC   $8905
8915-   8C 55 D6   STY   $D655
8918-   2C 88 C0   BIT   $C088
891B-   68         PLA
891C-   A8         TAY
891D-   20 B7 00   JSR   $00B7
8920-   F0 05      BEQ   $8927
8922-   20 B1 00   JSR   $00B1
8925-   D0 FB      BNE   $8922
8927-   60         RTS
8928-   8A         TXA
8929-   48         PHA
892A-   98         TYA
892B-   48         PHA
892C-   A2 00      LDX   #$00
892E-   86 85      STX   $85
8930-   A0 01      LDY   #$01
8932-   18         CLC
8933-   BD 00 02   LDA   $0200,X
8936-   F0 32      BEQ   $896A
8938-   B9 55 D6   LDA   $D655,Y
893B-   C9 5F      CMP   #$5F
893D-   F0 22      BEQ   $8961
893F-   C9 5B      CMP   #$5B
8941-   B0 19      BCS   $895C
8943-   C9 41      CMP   #$41
8945-   90 15      BCC   $895C
8947-   DD 00 02   CMP   $0200,X
894A-   F0 15      BEQ   $8961
894C-   09 20      ORA   #$20
894E-   DD 00 02   CMP   $0200,X
8951-   F0 0E      BEQ   $8961
8953-   A6 85      LDX   $85
8955-   E8         INX
8956-   86 85      STX   $85
8958-   A0 01      LDY   #$01
895A-   D0 D6      BNE   $8932
895C-   DD 00 02   CMP   $0200,X
895F-   D0 F2      BNE   $8953
8961-   CC 55 D6   CPY   $D655
8964-   B0 04      BCS   $896A
8966-   E8         INX
8967-   C8         INY
8968-   D0 C8      BNE   $8932
896A-   68         PLA
896B-   A8         TAY
896C-   68         PLA
896D-   AA         TAX
896E-   60         RTS
```

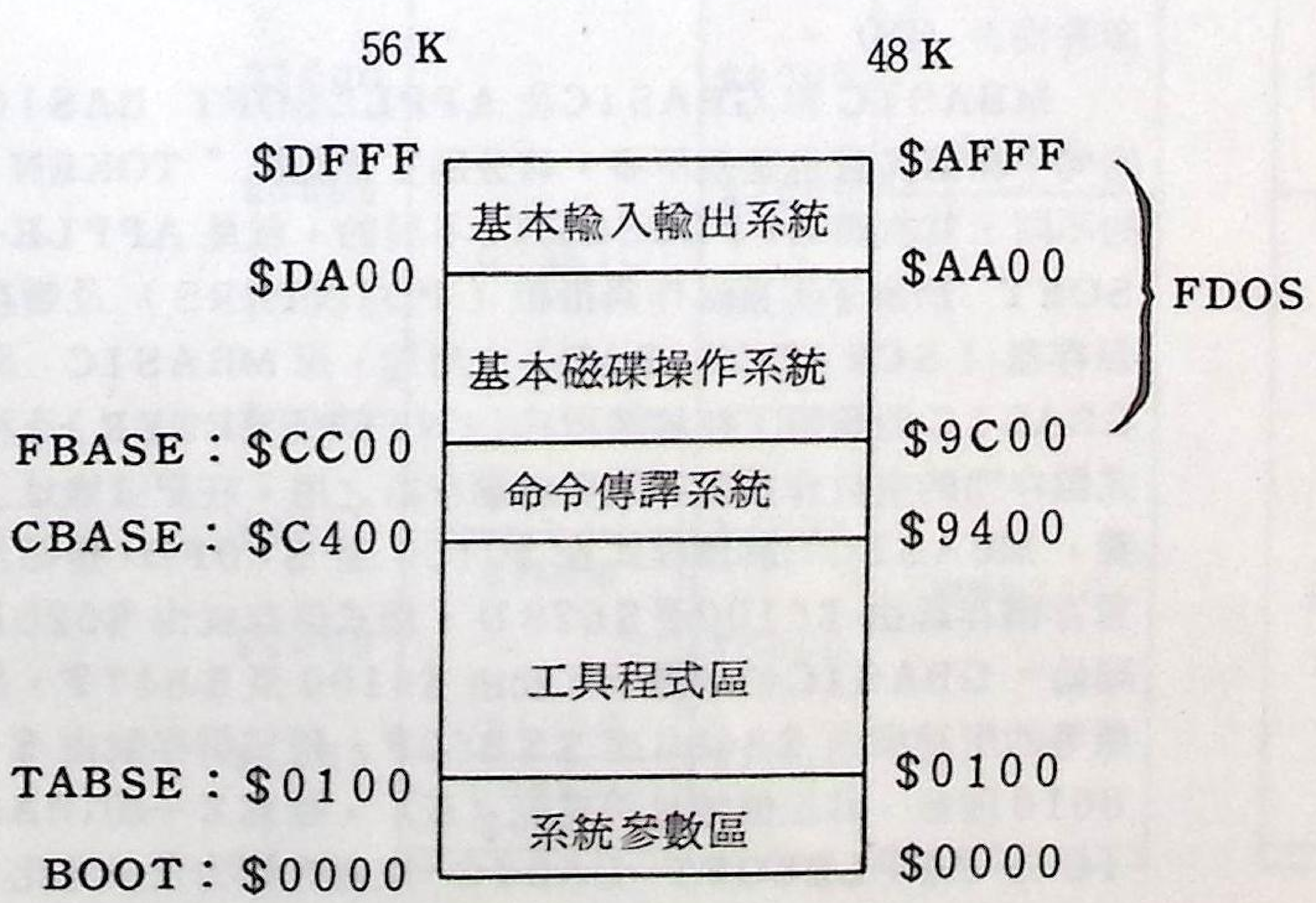
□

再談Z-80咭及其記憶地址配對法

由 MICROSOFT 研究成功的Z—80咭，把APPLE Ⅱ帶進了一個多彩多姿的領域，甚至可以說，APPLE Ⅱ 像脫胎換骨似的，變成了一個新的微電腦。微電腦用家都雀躍地研究如何把現行的6502軟件轉到Z—80的操作系統中，而他們之所以最爲根據的，全在於6502對Z—80的記憶地圖（MEMORY MAPPING）配對法。筆者在下面會詳盡地論述Z—80與6502在地址的配對法及用途。

現且先讓我們看看Z—80系統的心臟結構，它的操作系統叫做 CP／M，（CONTROL PROGRAM／MICRO —PROCESSOR），而 CP／M 可以說就是6502的機械語言系統（SYSTEM MONITOR）及磁碟操作系統的組成，CP／M爲用家提供了高階（HIGH LEVEL）的功能，如 BASIC 及對其他實用的用家程式，CP／M也提供了一些方法，令到這些高階的功能能夠使用到 CP／M 的資源。其資源就包括了系統的鍵盤，螢幕監視器，磁碟機，印字機或其他輸入及輸出的裝置。

圖（二）列出了 CP／M 的記憶地圖（MEMORY MAP），CP／M的操作系統就如6502的一樣，都是放在高地址的記憶區中，而與6502更相似的，就是它的零頁區都是被 SYSTEM 所預留了作爲系統的參數（SYSTEM PARAMETER）之用。CP／M大致上分爲五個組件，而它們全都是放在用家的讀寫記憶區（RAM）中，與6502不同的就是它沒有任何的程式是內置在ROM記憶體內。由於 CP／M 的特殊結構，在這五個組件中，有三個是不可以被 CP／M 系統所佔據的，它們分別爲基本輸入輸出系



圖（2）CP／M記憶地址圖

表（一）

組件名稱	48K系統		64K系統	
工具程式區	37K	44K	49K	56K
命令傳譯系統	2K		2K	
FDOS（BDOS＋BIOS）	5K		5K	
系統緩衝區	2K		2K	
輸入輸出區	4K		4K	
CP／M輸入輸出表	0.5K		0.5K	
6502叠層區	0.5K		0.5K	

（BIOS—BASIC I／O SYSTEM），基本磁碟操作系統（BDOS—BASIC DISK OPERATING SYSTEM）及零頁區，而BIOS 與 BDOS 合成一體又稱爲 FDOS 。其餘的組件爲命令傳譯系統（CCP—CONSOLE COMMAND PROCESSOR），負起了FDOS 間之傳譯工作，及工具程式區（TPA—TRANSIENT PROGRAM AREA），担任了程式的儲存及執行，所以又可以說是用家的記憶區，這記憶區的大小，又決定於用家的微電腦是否有語言咭（LANGUAGE CARD）或 RAM咭的裝置，由此說來，如果是全64K的 APPLE 微電腦，其所“BOOT”起的 CP／M 系統爲56K，而48K的 APPLE Ⅱ 微電腦，其所“BOOT”起的爲44K的CP／M系統，其關係可見表（一）及圖（二）。

當進入了 CP／M系統之後，6502 並不是完全靜寂的，正確地說，它變成了一個多微處理機系統（MULTI — PROCESSING SYSTEM)，這是因爲輸入及輸出的任務仍然交由6502處理，當Z—80需要讀取或存取文字幕、磁碟機、印字機或其他週邊裝置時，它把任務交給了6502代爲處理，當6502完成了所交來的任務時，控制便回到Z—80手中，要6502能夠完成由Z—80交來的任務，它的零頁區 ZERO PAGE），叠存區（STACK），文字幕記憶區（TEXT SCREEN MEMORY）及隨機輸入輸出緩衝區等便需要保存，又因爲基於這個原因，Z—80的記憶地址便一定要重新安排，以免兩個微處理器的記憶區相互混爲一體。

要達到這個目的，Z—80的零頁區便要移動至6502的記憶地址$1000 中，而6502的 $0000—$0FFF 便轉移至Z—80的 $F000—$FFFF中，而爲了能使6502的隨機輸入輸出地址完整起見，它由 $C000—$CFFF 間的地址便轉移至Z—80的$ E000—$EFFF中，它們相互的關係可參考表（三）。

在 APPLE Ⅱ Z—80中行使的 BASIC（MBASIC

程式（一）

```
10 REM ****************************************
20 REM *   BENCHMARK TESTS FOR APPLE II      *
30 REM * . MICROSOFT   - APPLESOFT BASIC      *
40 REM * . MICROSOFT   - GBASIC   (CP/M)      *
50 REM * . MICROSOFT   - MBASIC   (CP/M)      *
60 REM ****************************************

100 PRINT "START"
110 FOR K = 1 TO 1000
120 NEXT K
130 PRINT "END"
140 END
150 REM      BM1 - SIMPLE FOR/NEXT LOOP

200 PRINT "START"
210 K = 0
220 K = K + 1
230 IF K < 1000 THEN 220
240 PRINT "END"
250 END
260 REM        BM2 - IF/THEN LOOP

300 PRINT "START"
310 K = 0
320 K = K + 1
330 A = K / K + K - K
340 IF K < 1000 THEN 320
350 PRINT "END"
360 END
370 REM        BM3 - USE VARIABLES IN LINE 330

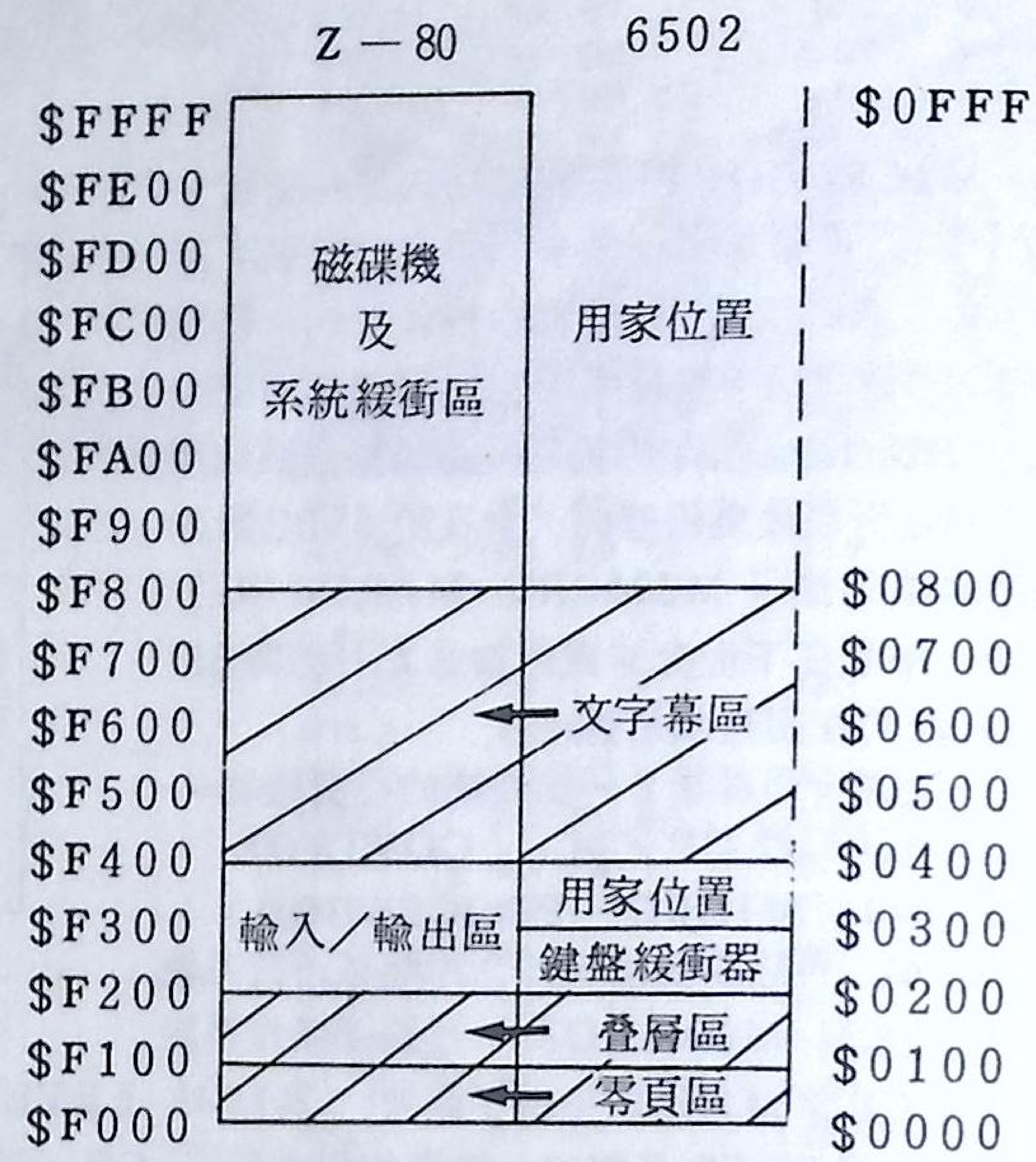
400 PRINT "START"
410 K = 0
420 K = K + 1
430 A = K / 2 * 3 + 4 - 5
440 IF K < 1000 THEN 420
450 PRINT "END"
460 END
470 REM        BM4 - USE VALUES IN LINE 430

500 PRINT "START"
510 K = 0
520 K = K + 1
530 A = K / 2 * 3 + 4 - 5
540 GOSUB 590
550 IF K < 1000 THEN 520
560 PRINT "END"
570 END
580 REM        BM5 - TEST GOSUB SPEED
590 RETURN     REM

600 PRINT "START"
610 K = 0
615 DIM M(5)
620 K = K + 1
630 A = K / 2 * 3 + 4 - 5
640 GOSUB 690
645 FOR L = 1 TO 5
650 IF K < 1000 THEN 620
660 PRINT "END"
670 END
680 REM        BM6 - ADD 2ND LOOP AND ARRAY M
690 RETURN : REM

700 PRINT "START"
710 K = 0
715 DIM M(5)
720 K = K + 1
730 A = K / 2 * 3 + 4 - 5
740 GOSUB 790
742 FOR L = 1 TO 5
744 NEXT L
750 IF K < 1000 THEN 720
760 PRINT "END"
770 END
780 REM        BM7 - ADD 744 TEST TIME TO STORE
                     VALUE IN ARRAY
790 RETURN : REM

800 PRINT "START"
810 K = 0
820 K = K + 1
830 A = K / 2
840 B = LOG (K)
850 C = SIN (K)
860 IF K Z 1000 THEN 820
870 PRINT "END"
880 END
890 REM BM8 - TEST BASIC FUNCTIONS
              AND EXPONENTATION
```



表（三） 低地址記憶地圖表

及GBASIC）都是在記憶地址＄0100開始載入，它們的分別只不過是GBASIC可以支持高解像區（HGR）而MBASIC則不能，而GBASIC則只在56K的CP／M系統磁碟才可以找到。在記憶容量方面，MBASIC佔據了24K的位置（APPLESOFT只需10K），GBASIC則佔據了32K的位置，由此算來，在56K的CP／M系統中，用家可以存取的記憶位置就只爲26—5K（MBASIC）及17.5（GBASIC），在44K的CP／M系統中，用家可以存取的位置則只有14.5K（MBASIC），以上的計算，是包括了CCP的位置在內，這是因爲CCP只在第一次〝BOOT〞才需要，以後的則轉成了用家存取地址，讀者可參看簡表（四）。

MBASIC與GBASIC及APPLESOFT BASIC的句子結構其實也是差不多，其分別主要在於〝TOKEN〞的不同，其次與APPLESOFT不同的，就是APPLE—SOFT預留了零頁區作爲指標（POINTERS）及暫存儲存器（SCRATCH PAD）之用途，而MBASIC及GBASIC則預留了在解譯程式（INTERPRETER）及程式儲存間的空位作爲指標及暫存儲存器之用，在記憶地址上看，MBASIC的解譯程式在＄0100至＄60FD，指標及暫存儲存器由＄6100至＄628D，程式儲存就由＄628E開始。GBASIC的解譯程式就由＄0100至＄847F，指標等的用途則由＄8480至＄＄860F，程式儲存就由＄8610開始，其記憶地址可見表（五），雖然Z—80 BASIC比APPLESOFT BASIC來得有威力及多元化，但在程式執行速度上却與APPLESOFT差不多，表（六）爲它們的定標點試驗結果，而定標點試驗程式則見程式（一）。

再談Z-80咭及其記憶地址配對法

在 6502中的監督程式負起了機械語言的列出，查驗及移動轉變，雖然在 CP／M 中是沒有它的監督程式去提供這些功能，但它却有一個工具程式，名爲除錯封裝（DYN—AMIC DEBUGGING TOOL—DDT）去做相類似的工作。要執行 DDT，它先從磁碟讀出，然後載入記憶地址 $0100至 $1500 中，再轉移至 $BB00（56K）或$8B00（44K）中。因爲如此，TPA的兩端地址便被DDT所佔用了而不能利用DDT 來查驗，要達到這個目的，在 TPA 中的程序便要先行儲起，用家才可執行 DDT 的命令了。

這次概略地論述了Z—80的記憶地址配對法，希望能對讀者在發掘Z—80及 CP／M 的世界時能有所幫助及裨益。□

	MBASIC	用家位置	GBASIC	用家位置
44 K	24 K	14.5 K	——	——
56 K	24 K	26.5 K	32 K	17.5 K

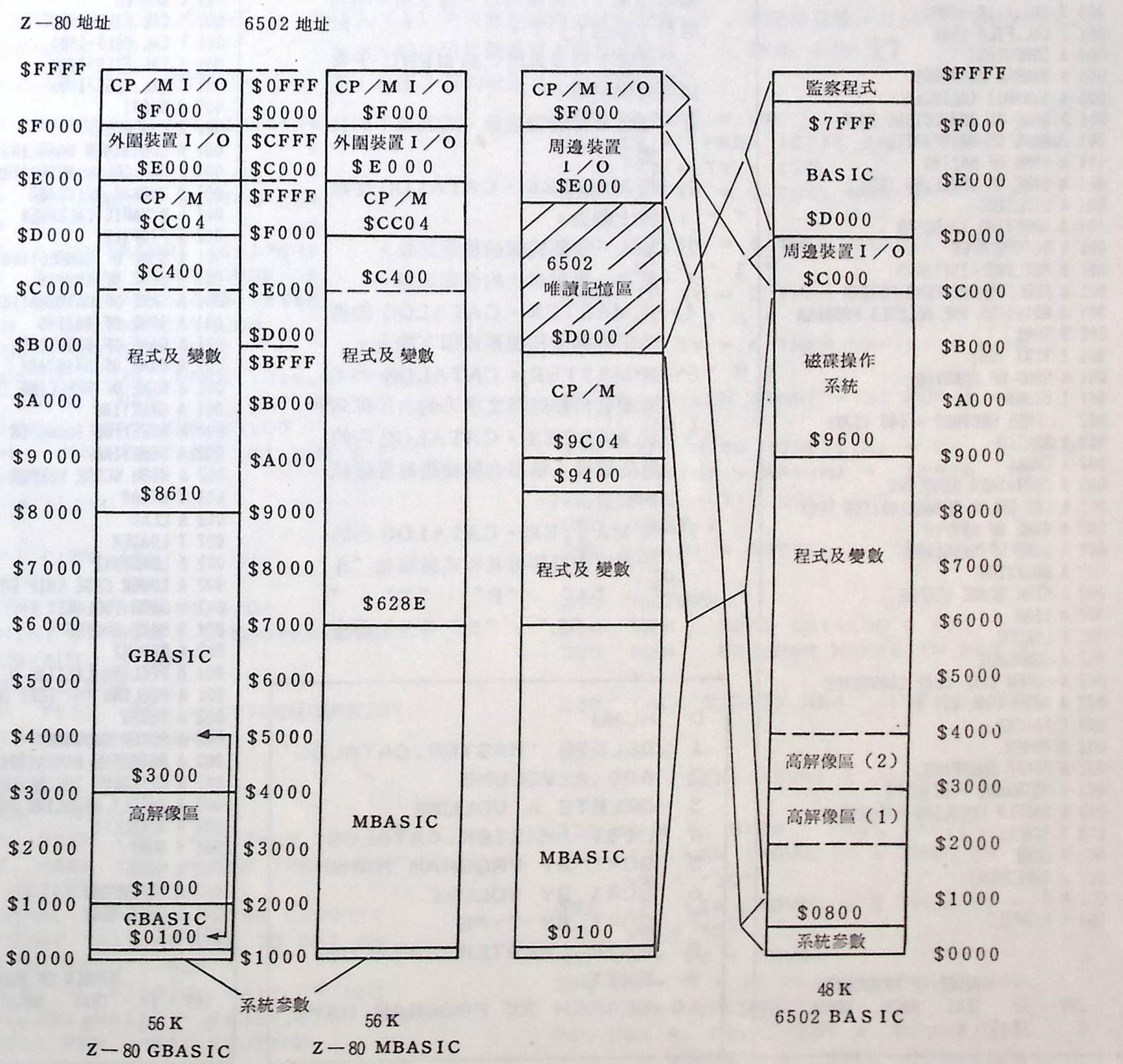
表（四） MBASIC及GBASIC記憶地址之比較

	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8
MBASIC	2.0	6.5	18.8	18.2	19.5	34.8	56.6	58.3
GBASIC	1.7	9.9	20.1	19.9	20.0	29.0	45.0	59.5

表（六） 定標點測討結果

註：單位爲秒

表（五） APPFE II Z—80記憶地圖



磁碟檔案黃頁系統

黃頁

本程式共有二個程式列表，它們在程式磁碟的 FILENAME 分別爲： CATALOG YELLOW PAGE SYSTEM 及 B—MASTER.CATALOG／AUTO—START。

相信閣下自從擁有磁碟機以來，肯定磁碟的數目與日俱增，對於磁碟內容很難全部記得，本應用程式的目的就是針對此而出發，希望將閣下手上的數以百計檔案滙編纂成一份方便目錄，好使一些功能相似的程式可以容易被找出。對擁有印字機的朋友來說會更方便，因爲他們可以把經過分析分類及索引的檔案印出來，以後便可以一目了然，不再需要「靠估！」了。

從表1 可見本程式的 MENU 上有10項可供使用。

0 —在各項功能完成後，回到圖1 的狀態下。

1 —把 MASTER・CATALOG 從磁碟上刪去。

2 —加入一隻新磁碟的檔案記錄。

3 —刪去一隻磁碟上的檔案記錄。

4 —將 MASTER・CATALOG 的檔案記錄列出往螢幕或印字機上。

5 —將 MASTER・CATALOG 內的檔案名稱按照英文字母的次序排列。

6 —將 MASTER・CATALOG 內的每隻磁碟上檔案名稱按照每隻磁碟的編號排列。

7 —將 MASTER・CATALOG 內的所有檔案把根據其程式種類如“A”，“I”，“B”，“T”，“R”，“L”，“S”等分類排序。

表①

```
0   MENU
1   DELETE 'MASTER.CATALOG'
2   ADD A VOLUME
3   DELETE A VOLUME
4   LIST 'MASTER.CATALOG'
5   SORT BY PROGRAM NAME
6   SORT BY VOLUME
7   SORT BY TYPE
8   SAVE 'MASTER.CATALOG'
9   EXIT
10  SEARCH BY PROGRAM NAME
```

圖（1）

```
VOL TYPE       PROGRAM
001     FREE SECTORS = 197 (49.25K)
001 A APPLE CALENDAR
001 A APPLE CALENDAR & SUM
001 A APPLESOFT TO FORTRAN
001 A BEND OF ROUNDING
001 B BOOLEAN TUTORIAL
001 A BOOLEAN TUTORIAL-MAIN PROGRAM
001 T CAL.FILE-1983
001 T CAL.FILE-1984
001 T CAL.FILE-1985
001 T CAL.FILE-1986
001 A COMPUTEXT
001 B DVORAK KEYBOARD
001 A DYNAMIC CALENDER
001 A GAME OF CONNECTING FOUR
001 A GAME OF MATHEMATIC
001 A GAME OF MAZING
001 A GAME OF SPELLING TEST
001 A GREETING
001 A GREETING TRANSFER
001 B MAZE SHAPES
001 B PEELING LISTING 5
001 A PEELING THE TEXT SCREEN
001 A RETRIEVE THE DELETED PROGRAM
001 B SUM1
001 T TEXT TEST
001 A GAME OF SABOTAGE
001 T HISAB
002     FREE SECTORS = 148 (37K)
002 B BPHTCH
002 B CHAR1
002 A CONTAINER HANDLING
002 A CRT COLOR DESMONTRATION TEST
002 A GAME OF CRYPTO
002 A GAME OF SABOTAGE
002 A GREETING
002 A HIGH SCORE KEEPER
002 A LEAD
002 T LOADER
002 A LOADSUM2
002 A LOWER CASE CHIP CONVERTER
002 A MATH TOOL KIT 1
002 A PATCH2
002 A POKER
002 A POKER SHOWHAND
002 A RETRIEVE FONTASTIC
002 A SAFELY HANDLING PROGRAM
002 B SCROLL!!
002 B SUM2
002 A SUPERFONT
002 A Z
002 T FIDFILE

            NUMBER OF PROGRAMS
 INT   FP   TEXT   MACH   TOTAL
 0     33   8      9      50
```

圖（2）

```
VOL TYPE       PROGRAM
002     FREE SECTORS = 148 (37K)
001     FREE SECTORS = 197 (49.25K)
001 A APPLE CALENDAR
001 A APPLE CALENDAR & SUM
001 A APPLESOFT TO FORTRAN
001 A BEND OF ROUNDING
001 B BOOLEAN TUTORIAL
001 A BOOLEAN TUTORIAL-MAIN PROGRAM
002 B BPHTCH
001 T CAL.FILE-1983
001 T CAL.FILE-1984
001 T CAL.FILE-1985
001 T CAL.FILE-1986
002 B CHAR1
001 A COMPUTEXT
002 A CONTAINER HANDLING
002 A CRT COLOR DESMONTRATION TEST
001 B DVORAK KEYBOARD
001 A DYNAMIC CALENDER
002 T FIDFILE
001 A GAME OF CONNECTING FOUR
002 A GAME OF CRYPTO
001 A GAME OF MATHEMATIC
001 A GAME OF MAZING
001 A GAME OF SABOTAGE
002 A GAME OF SABOTAGE
001 A GAME OF SPELLING TEST
001 A GREETING
001 A GREETING TRANSFER
002 A GREETING
002 A HIGH SCORE KEEPER
001 T HISAB
002 A LEAD
002 T LOADER
002 A LOADSUM2
002 A LOWER CASE CHIP CONVERTER
002 A MATH TOOL KIT 1
001 B MAZE SHAPES
002 A PATCH2
001 B PEELING LISTING 5
001 A PEELING THE TEXT SCREEN
002 A POKER
002 A POKER SHOWHAND
002 A RETRIEVE FONTASTIC
001 A RETRIEVE THE DELETED PROGRAM
002 A SAFELY HANDLING PROGRAM
002 B SCROLL!!
001 B SUM1
002 B SUM2
002 A SUPERFONT
001 T TEXT TEST
002 A Z

            NUMBER OF PROGRAMS
 INT   FP   TEXT   MACH   TOTAL
 0     33   8      9      50
```

磁碟檔案黃頁系統

8—將電腦內記憶體的內容，不論是否經過加添、刪除、修改等步驟，貯回磁碟上，檔名是〝MASTER·CATALOG”，閣下可以任意用其它名字來代替 MASTER·CATALOG。

註：MASTER·CATALOG 是一個文字檔案，內貯存一切檔案名稱的索引記錄。

9—離開本程式。

10—快速找尋程式名稱在那一隻磁碟上。

爲方便說明起見，以下我將用本刊第一二期程式磁碟來作一實例說明如何將磁碟上的檔案作索引分類：

①從 MENU 上選擇（2），當問及磁碟編號時，因爲是第一隻磁碟因此按〝001〞，將第一隻磁碟插入磁碟機再按〔RET〕，磁碟編號可以任意輸入，不過若果與以前所輸入的編號相同，便會出現兩個號碼相同的索引結果。當電腦將檔案名字讀入之後，便會回到 MENU 去。這時可重覆上述步驟把其他磁碟檔案名作索引分類。參閱圖 1。

②若果因爲意外按錯了磁碟編號而希望改正時，可以（3），按入錯誤的磁碟編號、〔RET〕，再按〔CTRL Y〕執行。

③按入（4）可以將記憶內的索引記錄列出。圖 1 所見便是將本刊程式磁碟第 1 及第 2 期的檔案名字依磁碟編號次序排列出後，系統清楚列出每類檔案的數目及總數。

④（5）（6）（7）這三項都是排序之用，功能則各有不同。圖 2 所見的例子便是將檔案以字母次序排列。

⑤若果閣下認爲要妥當的話可以按（8）以喜歡的檔案名字或按〔RETURN〕以〝MASTER·CATALOG” 爲檔案名貯存起來。

⑥若要找尋某個程式在那一隻磁碟上，可按（10），然後打入程式的檔案名字。

必須留意，本系統只能對沒有保護的磁碟有效，且必須爲 DOS 3.2 或 DOS 3.3。□

```
1  REM  ***********************
2  REM  *                     *
4  REM  * CATALOG YELLOW      *
5  REM  * PAGES   SYSTEM      *
6  REM  *                     *
7  REM  * COPYRIGHT (C)1984*
8  REM  *   BY COMPUHOUSE     *
9  REM  ***********************
10  REM   MASTER CATALOG
30  HIMEM: 20736: TEXT
40  POKE 1013,76: POKE 1014,0: POKE
    1015,81: REM      SET UP '&'
     HOOK. $3F5:JMP $5100
50  REM   TN$ MUST BE 1ST VAR & L
    ENGTH 30
60 TN$ = "MENU
          "
70 D$ =  CHR$ (4)
80 N =  - 1
90  PRINT D$;"NOMONC,I,O"
100  PRINT D$"BLOAD B.MASTER.CATA
    LOG/AUTO-START"
110  CALL 21504
120  IF  PEEK (28) = 1 THEN  PRINT
    "TN$ ERROR": END
130  DIM NA$(700)
140  GOTO 1000
150  REM   MOVE NAME
160  IF  PEEK (28) = 0 THEN 190
170  IF  PEEK (28) = 255 THEN  POP
    : RETURN
180  NORMAL : PRINT "DISK ERROR":
     PRINT "HIT RETURN TO GET BA
    CK TO MENU": POP
181  POKE  - 16368,0: WAIT  - 163
    84,128: WAIT  - 16384,1,1: GOTO
    1000: REM  WAIT ROUTINE
190 N = N + 1
200 NA$(N) = V$
210 TY =  PEEK (27): IF TY > 4 THEN
    TY = TY - 128
220  IF TY = 0 THEN NA$(N) = NA$(
    N) + " T "
230  IF TY = 1 THEN NA$(N) = NA$(
    N) + " I "
240  IF TY = 2 THEN NA$(N) = NA$(
    N) + " A "
250  IF TY = 4 THEN NA$(N) = NA$(
    N) + " B "
260 L =  LEN (TN$) - 3: FOR I = 1
     TO L
270  IF  MID$ (TN$,I,3) = "   " THEN
    NA$(N) = NA$(N) +  LEFT$ (TN
    $,I - 1): GOTO 300
280  NEXT I
290 NA$(N) = NA$(N) +  LEFT$ (TN$
    ,30)
300  RETURN
310  REM   READ CATALOG & PUT
320  REM   PROGRAM NAMES IN NA$(X
    X)
330  CALL 21507: REM   1ST SECTOR

332 V$ = ""
333  IF  LEN (ADV$) < > 0 THEN 3
    36
335 ADV$ =  STR$ ( PEEK (26))
336  IF  LEN (ADV$) = 1 THEN V$ =
    V$ + "00"
337  IF  LEN (ADV$) = 2 THEN V$ =
    V$ + "0"
339 V$ = V$ + ADV$
350 :N = N + 1: FLASH : PRINT "AD
    DING VOLUME ";V$
360 FS% =  PEEK (29) +  PEEK (30)
     * 256:FS$ =  STR$ (FS%):DF$
```

```
     =  STR$ (FS% / 4): IF  LEN
    (FS$) < 3 THEN FS$ =  LEFT$
    ("  ",3 -  LEN (FS$)) + FS$
370 NA$(N) = V$ + "    FREE SECTO
    RS = " + FS$ + " (" + DF$ +
    "K)"
380  GOSUB 150
390  CALL 21510: REM   LOOK FOR M
    ORE
400  GOSUB 150
410  GOTO 390
1000  NORMAL : HOME
1005  PRINT D$;"PR#0"
1008  HOME
1010  HTAB 7: PRINT "IMPROVED MAS
    TER CATALOG DOS 3.3"
1020  HTAB 10: INVERSE : PRINT "M
    ASTER CATALOG SYSTEM": NORMAL

1030  HTAB 15: PRINT "BY ALBERT N
    G"
1040  VTAB 6: HTAB 10: PRINT "OPT
    IONS"
1050  PRINT
1060  HTAB 5: PRINT "0  MENU"
1070  HTAB 5: PRINT "1  DELETE 'M
    ASTER.CATALOG'"
1080  HTAB 5: PRINT "2  ADD A VOL
    UME"
1090  HTAB 5: PRINT "3  DELETE A
    VOLUME"
1100  HTAB 5: PRINT "4  LIST 'MAS
    TER.CATALOG'"
1110  HTAB 5: PRINT "5  SORT BY P
    ROGRAM NAME"
1120  HTAB 5: PRINT "6  SORT BY V
    OLUME"
1130  HTAB 5: PRINT "7  SORT BY T
    YPE"
1140  HTAB 5: PRINT "8  SAVE 'MAS
    TER.CATALOG'"
1150  HTAB 5: PRINT "9  EXIT"
1155  HTAB 5: PRINT "10 SEARCH BY
    PROGRAM NAME"
1160  VTAB 19
1170  PRINT : INPUT "OPTION ";OP
1180  IF OP < 0 OR OP > 10 THEN 1
    160
1190  IF OP = 0 THEN 1000
1200  ON OP GOTO 1500,2000,3000,4
    000,5000,6000,9000,7000,8000
    ,3500
1500  REM      INITIALIZE MASTER.C
    AT
1501  HOME : INPUT "WHAT FILE NAM
    E? (DEFAULT NAME IS MASTER.C
    ATALOG)  ";FQ$
1502  PRINT
1503  IF FQ$ = "" THEN 1510
1504  GOTO 1515
1510  PRINT "TO DELETE MASTER.CAT
    ALOG"
1512  GOTO 1520
1515  PRINT "TO DELETE ";FQ$
1520  PRINT "ENTER '<CONTROL> Y';
    ELSE RETURN"
1530  GET A$
1540  IF A$ <  >  CHR$ (25) THEN
    1000
1550  PRINT
1560  ONERR  GOTO 1590
1563  IF FQ$ = "" THEN 1570
1564  PRINT D$;"DELETE ";FQ$;",VO
    "
1565  GOTO 1000
1570  PRINT D$"DELETE MASTER.CATA
    LOG,VO"
1580  GOTO 1000
1590  POKE 216,0:X =  PEEK (222)
1600  IF X = 6 THEN 1000
1610  RESUME
2000  REM    ADD A VOLUME
2003  PRINT
2005  INVERSE : PRINT "  INSERT
    DISK TO BE ADDED TO CATALOG
    ": NORMAL
2010  PRINT : PRINT " FOR DOS EN
    CODED VOLUME #'S ENTER '0'"
2020  PRINT : INPUT "      EN
    TER VOLUME # ";ADV$
2040  POKE (28),0
2050  GOSUB 310: REM   READ CATA
    LOG
2060 X = X + 1: IF X -  INT (X /
    5) * 5 = 0 THEN F =  FRE (0)
2070  GOTO 1000
3000  REM   DELETE A VOLUME
3010  PRINT
3020  INPUT "3 DIGIT VOLUME NUMBE
    R ";VD$
3030  IF  LEN (VD$) < 3 THEN VD$ =
     LEFT$ ("   ",3 -  LEN (VD$)
    ) + VD$
3032  HOME : INPUT "WHAT FILE NAM
    E TO DELETE THE VOL. IN?  (
    DEFAULT IS MASTER.CATALOG)";
    FQ$: PRINT
3034  IF FQ$ = "" THEN 3040
3036  PRINT D$;"OPEN ";FQ$;",VO"
3037  ONERR  GOTO 3120
3038  PRINT D$;"READ ";FQ$
3039  GOTO 3070
3040  PRINT D$;"OPEN MASTER.CATAL
    OG,VO"
3050  ONERR  GOTO 3120
3060  PRINT D$;"READ MASTER.CATAL
    OG"
3070 I =  - 1
3080 I = I + 1
```

```
3090  INPUT NA$(I)
3100  IF  LEFT$ (NA$(I),3) = VD$ THEN
   3090
3110  GOTO 3080
3120  POKE 216,0:X =  PEEK (222)
3130  IF X = 5 THEN  GOSUB 10000:
   N = I - 1: GOTO 7040
3140  RESUME
3500  REM   SEARCH BY PROGRAM NAM
   E
3510  HOME
3511  INPUT "DO YOU WANT PRINTOUT
    ? (DEFAULT=NO) ";ZZZ$: IF Z
   ZZ$ = "Y" THEN  PRINT D$"PR#
   1"
3512  IF ZZZ$ = "Y" THEN  PRINT  CHR$
   (15); CHR$ (27) +  CHR$ (65)
    +  CHR$ (9)
3520  INPUT "PROGRAM NAME?";PN$
3530  PRINT "(WORKING)"
3540 I = 1
3550  FOR I = 0 TO 2000
3560  IF  MID$ (NA$(I),7, LEN (PN
   $)) = PN$ THEN  PRINT NA$(I)
   : PRINT :FLAG = 1
3600  ONERR  GOTO 3700
3650  NEXT I
3699  PRINT
3700  IF FLAG = 1 THEN FLAG = 0: GOTO
   3800
3710  PRINT PN$;" NOT FOUND"
3799  PRINT
3800  INPUT "SEARCH FOR ANOTHER N
  AME? (Y/N)";YN$
3810  IF YN$ = "Y" THEN 3500
3820  GOTO 1000
4000  REM   LIST CATALOG
4010  IF N > 0 THEN 4110
4020  ONERR  GOTO 4080
4021  HOME : PRINT "THERE IS NO L
   IST IN MEMORY": PRINT
4023  INPUT "WHAT FILE NAME? (DEF
  AULT FILE NAME IS  MASTER.C
  ATALOG)";FQ$
4024  IF FQ$ = "" THEN  GOTO 4030
4026  PRINT D$;"OPEN ";FQ$;",VO"
4028  PRINT D$;"READ ";FQ$
4029  GOTO 4050
4030  PRINT D$;"OPEN MASTER.CATAL
  OG,VO"
4040  PRINT D$;"READ MASTER.CATAL
  OG"
4050 N = N + 1
4060  INPUT NA$(N)
4070  GOTO 4050
4080  POKE 216,0:X =  PEEK (222)
4090  IF X = 5 THEN  GOSUB 10000:
  N = N - 1: GOTO 4110
4100  RESUME
4110  HOME
4120 TI% = 0:TF% = 0:TT% = 0:TB% =
   0:TN% = 0
4130  INPUT "PRINTER (PR#1) ON (Y
  /N)? ";PT$
4132  IF PT$ <  > "Y" THEN 4139
4133  REM   TURN PRINTER ON
4134  PRINT D$;"PR#1"
4135  PRINT  CHR$ (27) CHR$ (67) CHR$
  (51)
4136  PRINT  CHR$ (27) CHR$ (78) CHR$
  (8)
4137  PRINT  CHR$ (27) CHR$ (65) CHR$
  (9)
4138  PRINT  CHR$ (15)
4139  PRINT "VOL TYPE      PROGR
  AM"
4140  IF N < 0 THEN 4270
4150  FOR I = 0 TO N
4160  PRINT NA$(I)
4170  IF  MID$ (NA$(I),5,1) = "I"
  THEN TI% = TI% + 1: GOTO 42
  20
4180  IF  MID$ (NA$(I),5,1) = "A"
  THEN TF% = TF% + 1: GOTO 42
  20
4190  IF  MID$ (NA$(I),5,1) = "T"
  THEN TT% = TT% + 1: GOTO 42
  20
4200  IF  MID$ (NA$(I),5,1) = " "
  THEN 4220
4210 TB% = TB% + 1
4220  IF  PEEK (37) < 21 THEN 426
  0
4225  IF PT$ = "Y" THEN 4255
4230  PRINT "'RETURN' FOR MORE.
  ESC' TO MENU"
4240  GET A$: IF A$ =  CHR$ (27) THEN
  1000
4245  IF A$ =  CHR$ (13) THEN  PRINT
  " CHR$(12):REM FORM FEED
4250  HOME : PRINT : GOTO 4260
4255  HOME
4260  NEXT I
4270 TN% = TI% + TF% + TT% + TB%
4280  PRINT : HTAB 11: PRINT "NUM
  BER OF PROGRAMS"
4290  PRINT "INT    FP    TEXT    MA
  CH   TOTAL"
4300  PRINT TI%; TAB( 7);TF%; TAB(
  14);TT%; TAB( 21);TB%; TAB(
  28);TN%
4303  IF PT$ = "N" THEN 4310
4304  REM   TURN OFF PRINTER
4305  PRINT D$;"PR#0"
4310  PRINT "RETURN TO MENU: 0"
4320  GOTO 1170
5000  REM   SORT BY PROGRAM NAME
5010  FLASH : PRINT "SORTING"
5020 N% = N
```

```
5030  & SRT#(NA$,0,N%,7,36,A,1,3,
     A)
5040  GOTO 1000
6000  REM   SORT BY VOLUME NO.
6010  FLASH : PRINT "SORTING"
6020 N% = N
6030  & SRT#(NA$,0,N%,1,3,A,7,36,
     A)
6040  GOTO 1000
7000  REM   SAVE CATALOG
7010  PRINT "INSERT 'MASTER.CATAL
     OG' DISKETTE"
7020  PRINT "AND HIT RETURN"
7030  GET A$
7040  PRINT
7050  ONERR  GOTO 7080
7051  HOME : INPUT "WHAT CATALOG
     FILE NAME TO BE SAVED UNDER?
      (DEFAULT NAME IS MASTER.CAT
     ALOG)";FQ$
7052  IF FQ$ = "" THEN 7060
7054  PRINT "THE NAME OF THE CAST
     ALOG FILE BEING SAVED IS ";F
     Q$
7055  PRINT
7057  PRINT D$;"DELETE ";FQ$;",VO
     "
7058  GOTO 7110
7060  PRINT D$;"DELETE MASTER.CAT
     ALOG,VO"
7070  GOTO 7110
7080  POKE 216,0:X =  PEEK (222)
7090  IF X = 6 THEN 7110
7100  RESUME
7110  IF N < 0 THEN 1000
7112  IF FQ$ = "" THEN 7120
7113  PRINT D$;"OPEN ";FQ$;",VO"
7114  PRINT D$;"WRITE ";FQ$
7115  GOTO 7140
7120  PRINT D$;"OPEN MASTER.CATAL
     OG,VO"
7130  PRINT D$;"WRITE MASTER.CATA
     LOG"
7140  FOR I = 0 TO N
7150  PRINT NA$(I)
7160  NEXT I
7162  IF FQ$ = "" THEN 7170
7163  PRINT D$;"CLOSE ";FQ$
7166  GOTO 1000
7170  PRINT D$;"CLOSE MASTER.CATA
     LOG"
7180  GOTO 1000
8000  REM   END
8010  POKE 216,0: END
9000  REM   SORT BY TYPE
9010  FLASH : PRINT "SORTING"
9020 N% = N
9030  & SRT#(NA$,0,N%,5,5,A,1,3,A
     ,7,36,A)
9040  GOTO 1000
10000  IF FQ$ = "" THEN 10009
10001  PRINT D$;"CLOSE ";FQ$
10002  RETURN
10009  PRINT D$;"CLOSE MASTER.CAT
     ALOG"
10010  RETURN
62999  REM         MASTER CATALOG
                   COPYRIGHT 1984 B
     Y COMPUTING AGE
63000  REM           MODIFIED BY ALB
     ERT NG
```

```
FILENAME:B.MASTER.CATALOG/
AUTOSTART, A$5100,L$45D
5100-   48          PHA
5101-   20 2C 53    JSR   $532C
5104-   68          PLA
5105-   A2 00       LDX   #$00
5107-   DD 72 53    CMP   $5372,X
510A-   D0 25       BNE   $5131
510C-   20 B1 00    JSR   $00B1
510F-   E8          INX
5110-   E0 05       CPX   #$05
5112-   D0 F3       BNE   $5107
5114-   A2 00       LDX   #$00
5116-   F0 03       BEQ   $511B
5118-   20 B1 00    JSR   $00B1
511B-   C9 2C       CMP   #$2C
511D-   F0 0A       BEQ   $5129
511F-   9D B8 53    STA   $53B8,X
5122-   E8          INX
5123-   E0 10       CPX   #$10
5125-   D0 F1       BNE   $5118
5127-   F0 08       BEQ   $5131
5129-   CA          DEX
512A-   BD B8 53    LDA   $53B8,X
512D-   C9 24       CMP   #$24
512F-   F0 03       BEQ   $5134
5131-   4C 84 51    JMP   $5184
5134-   A9 80       LDA   #$80
5136-   0D B9 53    ORA   $53B9
5139-   8D B9 53    STA   $53B9
513C-   A9 03       LDA   #$03
513E-   85 EC       STA   $EC
5140-   A9 00       LDA   #$00
5142-   85 F1       STA   $F1
5144-   20 B1 00    JSR   $00B1
```

```
5147-   20 67 DD    JSR   $DD67
514A-   20 52 E7    JSR   $E752
514D-   A5 50       LDA   $50
514F-   85 DE       STA   $DE
5151-   A5 51       LDA   $51
5153-   85 DF       STA   $DF
5155-   20 B1 00    JSR   $00B1
5158-   20 67 DD    JSR   $DD67
515B-   20 52 E7    JSR   $E752
515E-   A5 50       LDA   $50
5160-   85 D4       STA   $D4
5162-   18          CLC
5163-   69 01       ADC   #$01
5165-   85 E0       STA   $E0
5167-   A5 51       LDA   $51
5169-   85 D5       STA   $D5
516B-   69 00       ADC   #$00
516D-   85 E1       STA   $E1
516F-   A5 F1       LDA   $F1
5171-   D0 59       BNE   $51CC
5173-   F0 15       BEQ   $518A
5175-   A2 00       LDX   #$00
5177-   BD 77 53    LDA   $5377,X
517A-   09 80       ORA   #$80
517C-   20 ED FD    JSR   $FDED
517F-   E8          INX
5180-   E0 17       CPX   #$17
5182-   D0 F3       BNE   $5177
5184-   20 4F 53    JSR   $534F
5187-   4C C9 DE    JMP   $DEC9
518A-   A0 00       LDY   #$00
518C-   8C CF 53    STY   $53CF
518F-   20 B1 00    JSR   $00B1
5192-   20 F8 E6    JSR   $E6F8
5195-   CA          DEX
5196-   AC CF 53    LDY   $53CF
5199-   96 E2       STX   $E2,Y
519B-   20 B1 00    JSR   $00B1
519E-   20 F8 E6    JSR   $E6F8
51A1-   AC CF 53    LDY   $53CF
51A4-   96 E7       STX   $E7,Y
51A6-   20 B1 00    JSR   $00B1
51A9-   90 D9       BCC   $5184
51AB-   C9 44       CMP   #$44
51AD-   F0 04       BEQ   $51B3
51AF-   A9 FF       LDA   #$FF
51B1-   30 02       BMI   $51B5
51B3-   A9 00       LDA   #$00
51B5-   99 C8 53    STA   $53C8,Y
51B8-   C8          INY
51B9-   8C CF 53    STY   $53CF
51BC-   20 B1 00    JSR   $00B1
51BF-   C9 29       CMP   #$29
51C1-   F0 06       BEQ   $51C9
51C3-   C9 2C       CMP   #$2C
51C5-   F0 C8       BEQ   $518F
51C7-   D0 BB       BNE   $5184
51C9-   8C CE 53    STY   $53CE
51CC-   20 B1 00    JSR   $00B1
51CF-   D0 B3       BNE   $5184
51D1-   A0 00       LDY   #$00
51D3-   B1 6B       LDA   ($6B),Y
51D5-   CD B8 53    CMP   $53B8
51D8-   D0 08       BNE   $51E2
51DA-   C8          INY
51DB-   B1 6B       LDA   ($6B),Y
51DD-   CD B9 53    CMP   $53B9
51E0-   F0 2B       BEQ   $520D
51E2-   18          CLC
51E3-   A0 02       LDY   #$02
51E5-   B1 6B       LDA   ($6B),Y
51E7-   65 6B       ADC   $6B
51E9-   48          PHA
51EA-   C8          INY
51EB-   B1 6B       LDA   ($6B),Y
51ED-   65 6C       ADC   $6C
51EF-   85 6C       STA   $6C
51F1-   68          PLA
51F2-   85 6B       STA   $6B
51F4-   C5 6D       CMP   $6D
51F6-   A5 6C       LDA   $6C
51F8-   E5 6E       SBC   $6E
51FA-   B0 03       BCS   $51FF
51FC-   4C D1 51    JMP   $51D1
51FF-   A2 02       LDX   #$02
5201-   BD B8 53    LDA   $53B8,X
5204-   9D 81 53    STA   $5381,X
5207-   CA          DEX
5208-   10 F7       BPL   $5201
520A-   4C 75 51    JMP   $5175
520D-   18          CLC
520E-   A5 6B       LDA   $6B
5210-   69 07       ADC   #$07
5212-   85 52       STA   $52
5214-   A5 6C       LDA   $6C
5216-   69 00       ADC   #$00
5218-   85 53       STA   $53
521A-   A5 DE       LDA   $DE
521C-   85 50       STA   $50
521E-   A5 DF       LDA   $DF
5220-   85 51       STA   $51
5222-   A5 EC       LDA   $EC
5224-   85 54       STA   $54
5226-   A9 00       LDA   #$00
5228-   85 55       STA   $55
522A-   20 00 55    JSR   $5500
522D-   A5 50       LDA   $50
522F-   85 D6       STA   $D6
5231-   A5 51       LDA   $51
5233-   85 D7       STA   $D7
5235-   4C 45 52    JMP   $5245
5238-   18          CLC
5239-   A5 D6       LDA   $D6
523B-   65 EC       ADC   $EC
523D-   85 D6       STA   $D6
523F-   A5 D7       LDA   $D7
5241-   69 00       ADC   #$00
5243-   85 D7       STA   $D7
```

```
5245-    A0 01       LDY   #$01
5247-    B1 D6       LDA   ($D6),Y
5249-    85 D8       STA   $D8
524B-    C8          INY
524C-    B1 D6       LDA   ($D6),Y
524E-    85 D9       STA   $D9
5250-    18          CLC
5251-    A5 D6       LDA   $D6
5253-    65 EC       ADC   $EC
5255-    85 DA       STA   $DA
5257-    A5 D7       LDA   $D7
5259-    69 00       ADC   #$00
525B-    85 DB       STA   $DB
525D-    18          CLC
525E-    A5 DE       LDA   $DE
5260-    69 01       ADC   #$01
5262-    85 ED       STA   $ED
5264-    A5 DF       LDA   $DF
5266-    69 00       ADC   #$00
5268-    85 EE       STA   $EE
526A-    4C 7A 52    JMP   $527A
526D-    18          CLC
526E-    A5 DA       LDA   $DA
5270-    65 EC       ADC   $EC
5272-    85 DA       STA   $DA
5274-    A5 DB       LDA   $DB
5276-    69 00       ADC   #$00
5278-    85 DB       STA   $DB
527A-    A0 01       LDY   #$01
527C-    B1 DA       LDA   ($DA),Y
527E-    85 DC       STA   $DC
5280-    C8          INY
5281-    B1 DA       LDA   ($DA),Y
5283-    85 DD       STA   $DD
5285-    A0 00       LDY   #$00
5287-    B1 D6       LDA   ($D6),Y
5289-    F0 52       BEQ   $52DD
528B-    85 EF       STA   $EF
528D-    B1 DA       LDA   ($DA),Y
528F-    F0 4C       BEQ   $52DD
5291-    85 F0       STA   $F0
5293-    A2 00       LDX   #$00
5295-    B4 E2       LDY   $E2,X
5297-    BD C8 53    LDA   $53C8,X
529A-    30 0C       BMI   $52A8
529C-    B1 D8       LDA   ($D8),Y
529E-    D1 DC       CMP   ($DC),Y
52A0-    B0 14       BCS   $52B6
52A2-    20 07 53    JSR   $5307
52A5-    4C DD 52    JMP   $52DD
52A8-    B1 D8       LDA   ($D8),Y
52AA-    D1 DC       CMP   ($DC),Y
52AC-    90 2F       BCC   $52DD
52AE-    F0 19       BEQ   $52C9
52B0-    20 07 53    JSR   $5307
52B3-    4C DD 52    JMP   $52DD
52B6-    D0 25       BNE   $52DD
52B8-    C8          INY
52B9-    C4 EF       CPY   $EF
```

```
52BB-    F0 06       BEQ   $52C3
52BD-    C4 F0       CPY   $F0
52BF-    F0 16       BEQ   $52D7
52C1-    90 0F       BCC   $52D2
52C3-    C4 F0       CPY   $F0
52C5-    90 E9       BCC   $52B0
52C7-    F0 0E       BEQ   $52D7
52C9-    C8          INY
52CA-    C4 EF       CPY   $EF
52CC-    F0 09       BEQ   $52D7
52CE-    C4 F0       CPY   $F0
52D0-    F0 DE       BEQ   $52B0
52D2-    98          TYA
52D3-    D5 E7       CMP   $E7,X
52D5-    D0 C0       BNE   $5297
52D7-    E8          INX
52D8-    EC CE 53    CPX   $53CE
52DB-    D0 B8       BNE   $5295
52DD-    E6 ED       INC   $ED
52DF-    D0 02       BNE   $52E3
52E1-    E6 EE       INC   $EE
52E3-    A5 ED       LDA   $ED
52E5-    C5 E0       CMP   $E0
52E7-    A5 EE       LDA   $EE
52E9-    E5 E1       SBC   $E1
52EB-    90 14       BCC   $5301
52ED-    E6 DE       INC   $DE
52EF-    D0 02       BNE   $52F3
52F1-    E6 DF       INC   $DF
52F3-    A5 DE       LDA   $DE
52F5-    C5 D4       CMP   $D4
52F7-    A5 DF       LDA   $DF
52F9-    E5 D5       SBC   $D5
52FB-    90 07       BCC   $5304
52FD-    20 4F 53    JSR   $534F
5300-    60          RTS
5301-    4C 6D 52    JMP   $526D
5304-    4C 38 52    JMP   $5238
5307-    A0 00       LDY   #$00
5309-    B1 D6       LDA   ($D6),Y
530B-    48          PHA
530C-    C8          INY
530D-    A5 D8       LDA   $D8
530F-    91 DA       STA   ($DA),Y
5311-    C8          INY
5312-    A5 D9       LDA   $D9
5314-    91 DA       STA   ($DA),Y
5316-    A5 DD       LDA   $DD
5318-    91 D6       STA   ($D6),Y
531A-    85 D9       STA   $D9
531C-    88          DEY
531D-    A5 DC       LDA   $DC
531F-    91 D6       STA   ($D6),Y
5321-    85 D8       STA   $D8
5323-    88          DEY
5324-    B1 DA       LDA   ($DA),Y
5326-    91 D6       STA   ($D6),Y
5328-    68          PLA
5329-    91 DA       STA   ($DA),Y
```

```
532B-   60          RTS
532C-   A2 00       LDX   #$00
532E-   B5 D0       LDA   $D0,X
5330-   9D 8E 53    STA   $538E,X
5333-   E8          INX
5334-   E0 22       CPX   #$22
5336-   D0 F6       BNE   $532E
5338-   A5 6B       LDA   $6B
533A-   8D B6 53    STA   $53B6
533D-   A5 6C       LDA   $6C
533F-   8D B7 53    STA   $53B7
5342-   A2 00       LDX   #$00
5344-   B5 50       LDA   $50,X
5346-   9D B0 53    STA   $53B0,X
5349-   E8          INX
534A-   E0 06       CPX   #$06
534C-   D0 F6       BNE   $5344
534E-   60          RTS
534F-   A2 00       LDX   #$00
5351-   BD 8E 53    LDA   $538E,X
5354-   95 D0       STA   $D0,X
5356-   E8          INX
5357-   E0 22       CPX   #$22
5359-   D0 F6       BNE   $5351
535B-   AD B6 53    LDA   $53B6
535E-   85 6B       STA   $6B
5360-   AD B7 53    LDA   $53B7
5363-   85 6C       STA   $6C
5365-   A2 00       LDX   #$00
5367-   BD B0 53    LDA   $53B0,X
536A-   95 50       STA   $50,X
536C-   E8          INX
536D-   E0 06       CPX   #$06
536F-   D0 F6       BNE   $5367
5371-   60          RTS
5372-   53          ???
5373-   52          ???
5374-   54          ???
5375-   23          ???
5376-   28          PLP
5377-   8D 56 41    STA   $4156
537A-   52          ???
537B-   49 41       EOR   #$41
537D-   42          ???
537E-   4C 45 20    JMP   $2045
5381-   20 20 20    JSR   $2020
5384-   20 4E 4F    JSR   $4F4E
5387-   54          ???
5388-   20 46 4F    JSR   $4F46
538B-   55 4E       EOR   $4E,X
538D-   44          ???
538E-   63          ???
538F-   10 FF       BPL   $5390
5391-   FF          ???
5392-   00          BRK
5393-   00          BRK
5394-   00          BRK
5395-   00          BRK
5396-   00          BRK
5397-   03          ???
5398-   C8          INY
5399-   47          ???
539A-   8B          ???
539B-   47          ???
539C-   6B          ???
539D-   DC          ???
539E-   E8          INX
539F-   B7          ???
53A0-   FF          ???
53A1-   FF          ???
53A2-   8B          ???
53A3-   47          ???
53A4-   DD 47 08    CMP   $0847,X
53A7-   08          PHP
53A8-   08          PHP
53A9-   00          BRK
53AA-   00          BRK
53AB-   00          BRK
53AC-   00          BRK
53AD-   00          BRK
53AE-   00          BRK
53AF-   01 88       ORA   ($88,X)
53B1-   13          ???
53B2-   55 52       EOR   $52,X
53B4-   00          BRK
53B5-   07          ???
53B6-   CB          ???
53B7-   1D 4E C1    ORA   $C14E,X
53BA-   24 00       BIT   $00
53BC-   00          BRK
53BD-   00          BRK
53BE-   00          BRK
53BF-   00          BRK
53C0-   00          BRK
53C1-   00          BRK
53C2-   00          BRK
53C3-   00          BRK
53C4-   00          BRK
53C5-   00          BRK
53C6-   00          BRK
53C7-   00          BRK
53C8-   FF          ???
53C9-   FF          ???
53CA-   00          BRK
53CB-   00          BRK
53CC-   00          BRK
53CD-   00          BRK
53CE-   02          ???
53CF-   02          ???
53D0-   00          BRK
53D1-   00          BRK
53D2-   00          BRK
53D3-   00          BRK
53D4-   00          BRK
53D5-   00          BRK
53D6-   00          BRK
53D7-   00          BRK
53D8-   00          BRK
```

```
53D9-   00          BRK
53DA-   00          BRK
53DB-   00          BRK
53DC-   00          BRK
53DD-   00          BRK
53DE-   00          BRK
53DF-   00          BRK
53E0-   00          BRK
53E1-   00          BRK
53E2-   00          BRK
53E3-   00          BRK
53E4-   00          BRK
53E5-   00          BRK
53E6-   00          BRK
53E7-   00          BRK
53E8-   00          BRK
53E9-   00          BRK
53EA-   00          BRK
53EB-   00          BRK
53EC-   00          BRK
53ED-   00          BRK
53EE-   00          BRK
53EF-   00          BRK
53F0-   00          BRK
53F1-   00          BRK
53F2-   00          BRK
53F3-   00          BRK
53F4-   00          BRK
53F5-   00          BRK
53F6-   00          BRK
53F7-   00          BRK
53F8-   00          BRK
53F9-   00          BRK
53FA-   00          BRK
53FB-   00          BRK
53FC-   00          BRK
53FD-   00          BRK
53FE-   00          BRK
53FF-   00          BRK
5400-   4C 09 54    JMP   $5409
5403-   4C 3D 54    JMP   $543D
5406-   4C A3 54    JMP   $54A3
5409-   A0 00       LDY   #$00
540B-   84 1C       STY   $1C
540D-   B1 69       LDA   ($69),Y
540F-   C9 54       CMP   #$54
5411-   D0 0E       BNE   $5421
5413-   C8          INY
5414-   B1 69       LDA   ($69),Y
5416-   C9 CE       CMP   #$CE
5418-   D0 07       BNE   $5421
541A-   C8          INY
541B-   B1 69       LDA   ($69),Y
541D-   C9 1E       CMP   #$1E
541F-   F0 05       BEQ   $5426
5421-   A9 01       LDA   #$01
5423-   85 1C       STA   $1C
5425-   60          RTS
5426-   C8          INY
5427-   B1 69       LDA   ($69),Y
5429-   85 18       STA   $18
542B-   C8          INY
542C-   B1 69       LDA   ($69),Y
542E-   85 19       STA   $19
5430-   20 E3 03    JSR   $03E3
5433-   84 E0       STY   $E0
5435-   85 E1       STA   $E1
5437-   60          RTS
5438-   A9 FF       LDA   #$FF
543A-   85 1C       STA   $1C
543C-   60          RTS
543D-   A0 03       LDY   #$03
543F-   A9 00       LDA   #$00
5441-   91 E0       STA   ($E0),Y
5443-   C8          INY
5444-   C8          INY
5445-   91 E0       STA   ($E0),Y
5447-   88          DEY
5448-   A9 11       LDA   #$11
544A-   91 E0       STA   ($E0),Y
544C-   20 C5 54    JSR   $54C5
544F-   A9 00       LDA   #$00
5451-   85 1D       STA   $1D
5453-   85 1E       STA   $1E
5455-   A0 38       LDY   #$38
5457-   B9 00 56    LDA   $5600,Y
545A-   A2 07       LDX   #$07
545C-   0A          ASL
545D-   90 06       BCC   $5465
545F-   E6 1D       INC   $1D
5461-   D0 02       BNE   $5465
5463-   E6 1E       INC   $1E
5465-   CA          DEX
5466-   10 F4       BPL   $545C
5468-   C8          INY
5469-   C0 C4       CPY   #$C4
546B-   90 EA       BCC   $5457
546D-   A0 0E       LDY   #$0E
546F-   B1 E0       LDA   ($E0),Y
5471-   85 1A       STA   $1A
5473-   A2 01       LDX   #$01
5475-   BD 00 56    LDA   $5600,X
5478-   30 BE       BMI   $5438
547A-   A0 04       LDY   #$04
547C-   91 E0       STA   ($E0),Y
547E-   E8          INX
547F-   BD 00 56    LDA   $5600,X
5482-   C8          INY
5483-   91 E0       STA   ($E0),Y
5485-   20 C5 54    JSR   $54C5
5488-   A2 0B       LDX   #$0B
548A-   BD 00 56    LDA   $5600,X
548D-   F0 A9       BEQ   $5438
548F-   30 2F       BMI   $54C0
5491-   C9 23       CMP   #$23
5493-   B0 A3       BCS   $5438
5495-   E8          INX
5496-   E8          INX
5497-   BD 00 56    LDA   $5600,X
549A-   85 1B       STA   $1B
```

```
549C-    8E F5 54    STX   $54F5
549F-    20 E5 54    JSR   $54E5
54A2-    60          RTS
54A3-    AE F5 54    LDX   $54F5
54A6-    8A          TXA
54A7-    18          CLC
54A8-    69 21       ADC   #$21
54AA-    AA          TAX
54AB-    B0 03       BCS   $54B0
54AD-    4C 8A 54    JMP   $548A
54B0-    AD 01 56    LDA   $5601
54B3-    A0 04       LDY   #$04
54B5-    91 E0       STA   ($E0),Y
54B7-    C8          INY
54B8-    AD 02 56    LDA   $5602
54BB-    91 E0       STA   ($E0),Y
54BD-    4C 85 54    JMP   $5485
54C0-    E8          INX
54C1-    E8          INX
54C2-    4C A6 54    JMP   $54A6
54C5-    A0 0C       LDY   #$0C
54C7-    A9 01       LDA   #$01
54C9-    91 E0       STA   ($E0),Y
54CB-    A0 08       LDY   #$08
54CD-    AD F6 54    LDA   $54F6
54D0-    91 E0       STA   ($E0),Y
54D2-    C8          INY
54D3-    AD F7 54    LDA   $54F7
54D6-    91 E0       STA   ($E0),Y
54D8-    20 E3 03    JSR   $03E3
54DB-    20 D9 03    JSR   $03D9
54DE-    90 04       BCC   $54E4
54E0-    84 1C       STY   $1C
54E2-    68          PLA
54E3-    68          PLA
54E4-    60          RTS
54E5-    A0 00       LDY   #$00
54E7-    E8          INX
54E8-    BD 00 56    LDA   $5600,X
54EB-    29 7F       AND   #$7F
54ED-    91 18       STA   ($18),Y
54EF-    C8          INY
54F0-    C0 1E       CPY   #$1E
54F2-    D0 F3       BNE   $54E7
54F4-    60          RTS
54F5-    76 00       ROR   $00,X
54F7-    56 00       LSR   $00,X
54F9-    00          BRK
54FA-    00          BRK
54FB-    00          BRK
54FC-    00          BRK
54FD-    00          BRK
54FE-    00          BRK
```

```
54FF-    00          BRK
5500-    A0 10       LDY   #$10
5502-    A5 50       LDA   $50
5504-    4A          LSR
5505-    90 0C       BCC   $5513
5507-    18          CLC
5508-    A2 FE       LDX   #$FE
550A-    B5 54       LDA   $54,X
550C-    75 56       ADC   $56,X
550E-    95 54       STA   $54,X
5510-    E8          INX
5511-    D0 F7       BNE   $550A
5513-    A2 03       LDX   #$03
5515-    76 50       ROR   $50,X
5517-    CA          DEX
5518-    10 FB       BPL   $5515
551A-    88          DEY
551B-    D0 E5       BNE   $5502
551D-    60          RTS
551E-    20 41 55    JSR   $5541
5521-    A0 10       LDY   #$10
5523-    06 50       ASL   $50
5525-    26 51       ROL   $51
5527-    26 52       ROL   $52
5529-    26 53       ROL   $53
552B-    38          SEC
552C-    A5 52       LDA   $52
552E-    E5 54       SBC   $54
5530-    AA          TAX
5531-    A5 53       LDA   $53
5533-    E5 55       SBC   $55
5535-    90 06       BCC   $553D
5537-    86 52       STX   $52
5539-    85 53       STA   $53
553B-    E6 50       INC   $50
553D-    88          DEY
553E-    D0 E3       BNE   $5523
5540-    60          RTS
5541-    A0 00       LDY   #$00
5543-    84 2F       STY   $2F
5545-    A2 54       LDX   #$54
5547-    20 4C 55    JSR   $554C
554A-    A2 50       LDX   #$50
554C-    B5 01       LDA   $01,X
554E-    10 0D       BPL   $555D
5550-    38          SEC
5551-    98          TYA
5552-    F5 00       SBC   $00,X
5554-    95 00       STA   $00,X
5556-    98          TYA
5557-    F5 01       SBC   $01,X
5559-    95 01       STA   $01,X
555B-    E6 2F       INC   $2F
555D-    60          RTS
```

□

未來領導科技潮流

集處理器、記憶、通訊於一身的

T424 Transputer

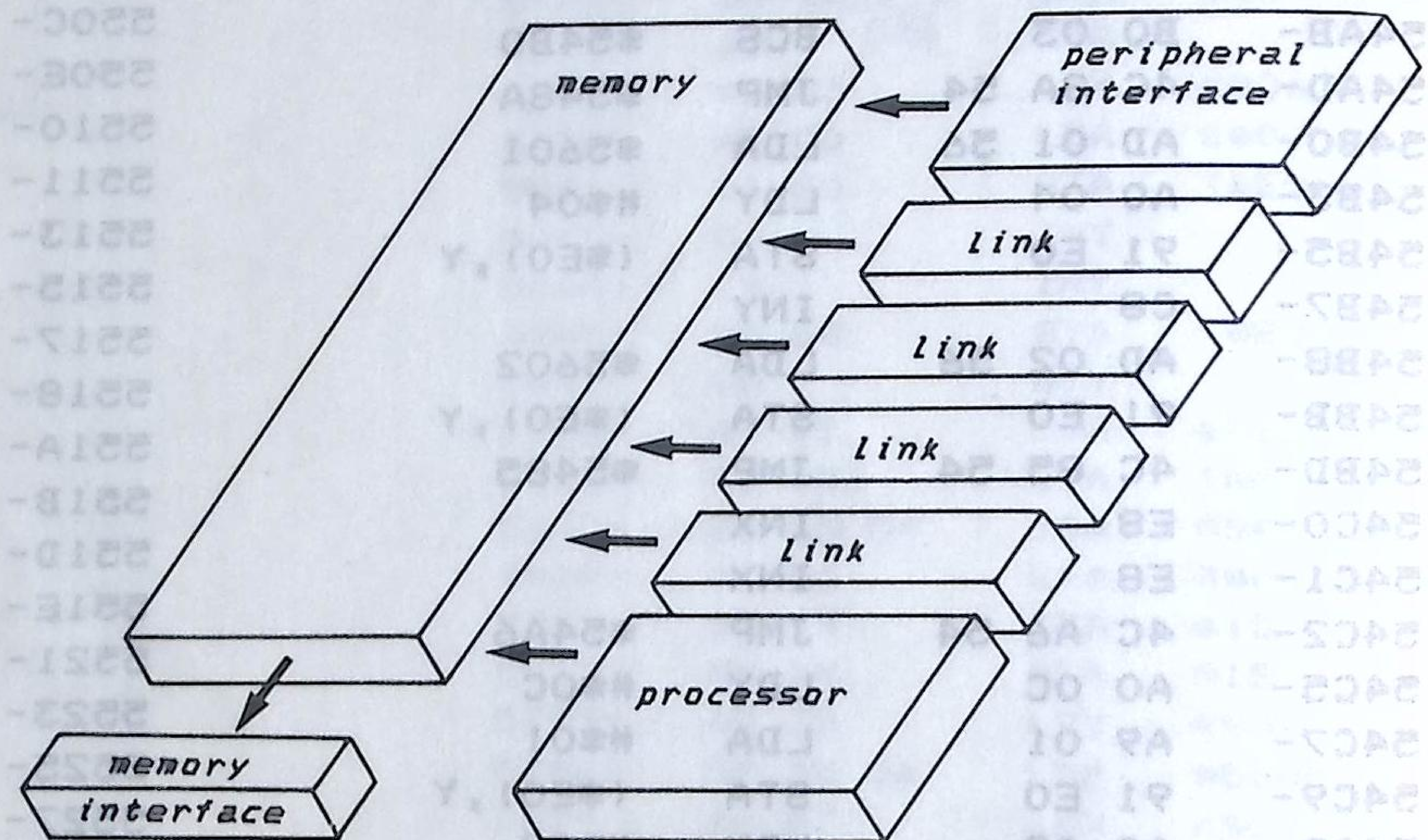
T424TRANSPUTER是革命性新的產品，英國現正努力研究，進度方面比美國及日本領先。雖然目前直接的投資研究費用還低於五百萬英鎊，但是廠房及研究組織的資金及貸款已超過一億英鎊。英國 VLSI 生產商IMMOS正躊躇滿志，爲新產品努力耕耘。

INMOS TRANSPUTER內微處理器，記憶系統，及通訊，全部都在一片四分之一平方吋晶片內。這產品不但可以節省了大量空間，亦不致減慢了處理時間。

四分之一平方吋晶片只是 64K BYTE 動態 RAM 的體積一半，而有 4KB記憶容量。這個整體組合的表現是10mIPS（每秒一千萬個指令）。INMOS已成功創造將三個元素設在一個晶片上的科技，只差接駁線路問題仍待克服。無可置疑，TRANSPUTER未來將成爲完整電腦系統的一個組合元件，INMOS提早宣佈TRANSPUTER的規格，只希望其他公司的大型及威力大的電腦系統在設計時能配合採用TR—ANSPUTER。

INMOS的總裁IANN BARR—ON 是創造TRANSPUTER主要人物。他的概念始於七十年代中期，而獲得發展及研究部撥款五百萬鎊後，四年前開始着手埋首鑽研。TRANSPUTER目標是爲第五代電腦系統而設計。日本方面亦已化費數十億美元於研究生產"人工智能"機器。而預料可於九十年代初期面世。當這消息公佈後，全球的電腦工業都產生焦慮，因爲他們都走同樣路綫，未來競爭眞是未知鹿死誰手。

現在你可以明白TRANSPUTER的重要性，因爲第五代電腦的晶片都一定全部包括處理，記憶及通訊功能。而TRANSPUTER在市場上已早着先鞭，英國的研究相信已比日本及美國領前了三年。



人工智能系統，或稱爲專家系統，主要是以規例爲基礎。電腦搜索這些規例而作出決定。當系統被"教導"後，就不會忘記任何細節，或任何重要的資料。不過如果電腦的資料庫沒存有適當的知識，它所作出的決定可能便會錯誤。

簡單而言，人工智能需要規例與知識——資訊存於儲料庫內，而當輸入指令後便可從電腦取得資訊。

但問題亦出於這裏，關鍵是：規例很少是線性及有次序的，它實是平衡及橫行。就好像玩迷宮遊戲一樣，某一階段有很多可能性，而要逐步逐步「解結」，要經過數百個程序才可找到解決方法。如果利用現行的電腦去尋找答案，搜索時間就會太久，令用者感到煩悶。

INMOS 所製造的晶片，就是希望配合未來的需要，令電腦可以平衡思想。OCCAM是程式語言，亦是工作規格語言（JOB SPECIFICATON LANGUAGE）。它同是設計及工作工具。它亦是爲並存機器（CONCUR—RENT MACHINE)而設計去發展

T424規格

● 32 BIT系統
10 MIPS（每秒可執行一千萬個指令。

● 處理器
同樣效能可減少指令數目
運算效率極高

● 記憶量
每晶片 4K BYTES
資料處理速度每秒 80 M BYTES

● 記憶介面
32 元數 MULTIPLEX 介面混合記憶系統。
直資料處理速度每秒 25 M BYTES

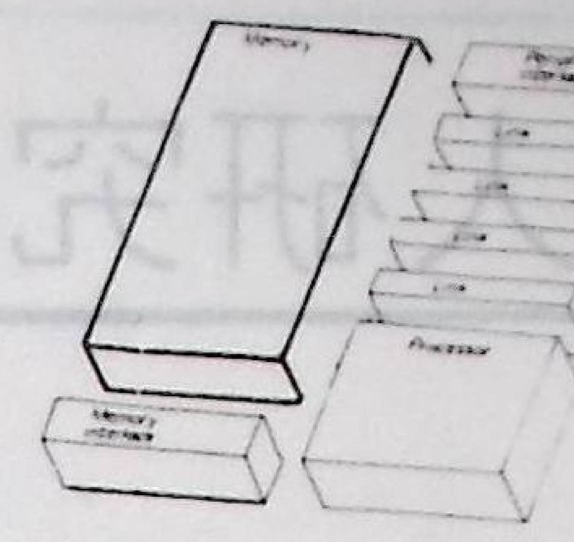
● 週邊介面
8元數 MULTIPLEX 介面
工業標準機器每秒 4 M BYTES

● INMOS 連繫
4個 INMOS 標準 TRANSPUT—ER 連繫
資料處理每秒1.5 M BYTES

● 技術
250,000 機器
2 MICRON CMOS

● 程式設計
可用標準高階語言寫程式
直接執行 OCCAM，極有效率，亦可並時進行。

Transputer



TRANSPUTER。

INMOS聲言一個TRANSPU—TER 的功能相等於一百部家庭電腦，並比目前最熱門的NATIONAL 32—BIT 處理器的速度快八倍。TRA—NSPUTER 已為平行系統（PARA—LLEL SYSTEM）開創了新領域，並大幅削減了電腦的價格及能源消耗。TRANSPUTER交流資料的速度非常高（1.5 M BYTE／SEC）。

INMOS晶片與其他晶片不一樣，它等於INTEL及MOTOROLA所生產的微處理器。INMOS 晶片倘若成功的話，它會在未來二十年內，領導整個潮流及電腦科技界，一如 IBM 的權威性。使用TRANSPUTER的數量並無限制，將來一個系統可以用上數以百計的TRANSPUTER，使威力大大增加。

T 424，是16—BIT TRAN—SPUTER，磁碟處理器，繪圖處理器，及所有標準介面，研究已有相當進展。

T424在八五年之前不會面世，初步它將打進高價值區域，如訊號處理。若研究有突破性的發展，八十年代末前，TRANSPUTER 將會出現於家庭產品上。

英倫 IMPERIAL COLLEGE 正研究 ALICE（APPLICATIVE LANGUAGE IDEALIZED CO—MPUTING ENGINE 應用語言理想化電算機器），是以 TRANSP—UTERS 組成的大型平衡電腦。此系統之所以重要是因為並不首先製造硬件。它是以"軟件"來設計，而硬件只淪為次要。設計組的首腦宣稱，"我們比日本進步了三年"，意思上當然亦表示比美國走前了一大步。他們所說也沒有誇大，因為到目前為止，地球上還未出現處理器，記憶容量及通訊三為一體的晶片。□

高空擲物 （上接第69頁）

```
1200  IF DD -  INT (DD / 10) * 10
      = 0 THEN DD = DD + 1: GOSUB
      4000
1205 KA =  PEEK ( - 16384): POKE
      - 16368,0
1210  IF KA = 136 THEN DD = DD +
      1: GOTO 900
1220  IF KA = 149 THEN DD = DD +
      1: GOTO 840
1230  RETURN
1300  DATA 6,0,14,0,75,0,129,0,14
      6,0,151,0,177,0,9,45,13,17,2
      7,63,63,23,9,45,13,17,27,27,
      31,19,9,45,13,49,59,63,63,19
      ,9,45,13,17,59,63,63,19,9,45
      ,13,17
1310  DATA 27,59,59,19,9,13,41,17
      ,59,27,59,19,45,13,9,21,59,2
      7,27,51,9,9,9,53,0,9,9,41,9,
      9,9,9,26,27,27,27,27,31,27,1
      9,9,9,41,9,9,9,9,26,27,27
1320  DATA 59,63,63,59,55,45,45,4
      5,45,45,9,9,62,63,63,63,63,6
      3,59,55,9,9,9,9,9,9,9,2,0,45
      ,9,9,17,63,63,63,55,45,9,9,1
      7,27,27,23,0,45,53,63,55
1330  DATA 0,41,9,9,9,21,63,31,27
      ,59,55,45,45,45,45,53,63,31,
      27,59,55,41,9,9,9,21,0,41,21
      ,59,23,45,53,59,23,41,21,59,
      23,41,17,0,999
1500  IF D THEN J = 4: GOTO 940
1505 MA = MA + 5
1510  GOSUB 940:DE = 5: GOSUB 200
      0: GOSUB 940:J = 0:NS = NS +
      1:GE = MA: GOTO 940
2000  FOR Z = 1 TO DE:LL =  PEEK
      (W) -  PEEK (W) +  PEEK (W):
      NEXT : RETURN
3000  HPLOT 0,115 TO 279,115
3010  HPLOT 8,92 TO 8,115: HPLOT
      200,92 TO 200,115: HPLOT 221
      ,92 TO 221,115
3020  HPLOT 6,92 TO 22,80 TO 39,9
      2: HPLOT 6,91 TO 22,79 TO 39
      ,91
3030  HPLOT 198,92 TO 210,80 TO 2
      23,92: HPLOT 198,91 TO 210,7
      9 TO 223,91
3040  HPLOT 37,91 TO 37,115
3045  HPLOT 208,90 TO 212,90 TO 2
      12,94 TO 208,94 TO 208,90
3050  RETURN
4000  IF D THEN D = 0: HCOLOR= 3:
      GOTO 4020
4010  HCOLOR= 0:D = 1
4020  HPLOT 199,97 TO 192,97 TO 1
      92,114
4030  HPLOT 194,105: HCOLOR= 3: RETURN

4060  PRINT "": PRINT "WANT TO PL
      AY AGAIN ? ";
4070  GET A$
4080  IF A$ = "Y" THEN  RUN
4090  IF A$ = "N" THEN  TEXT : HOME
      : END
5000  GOTO 4070
```

□

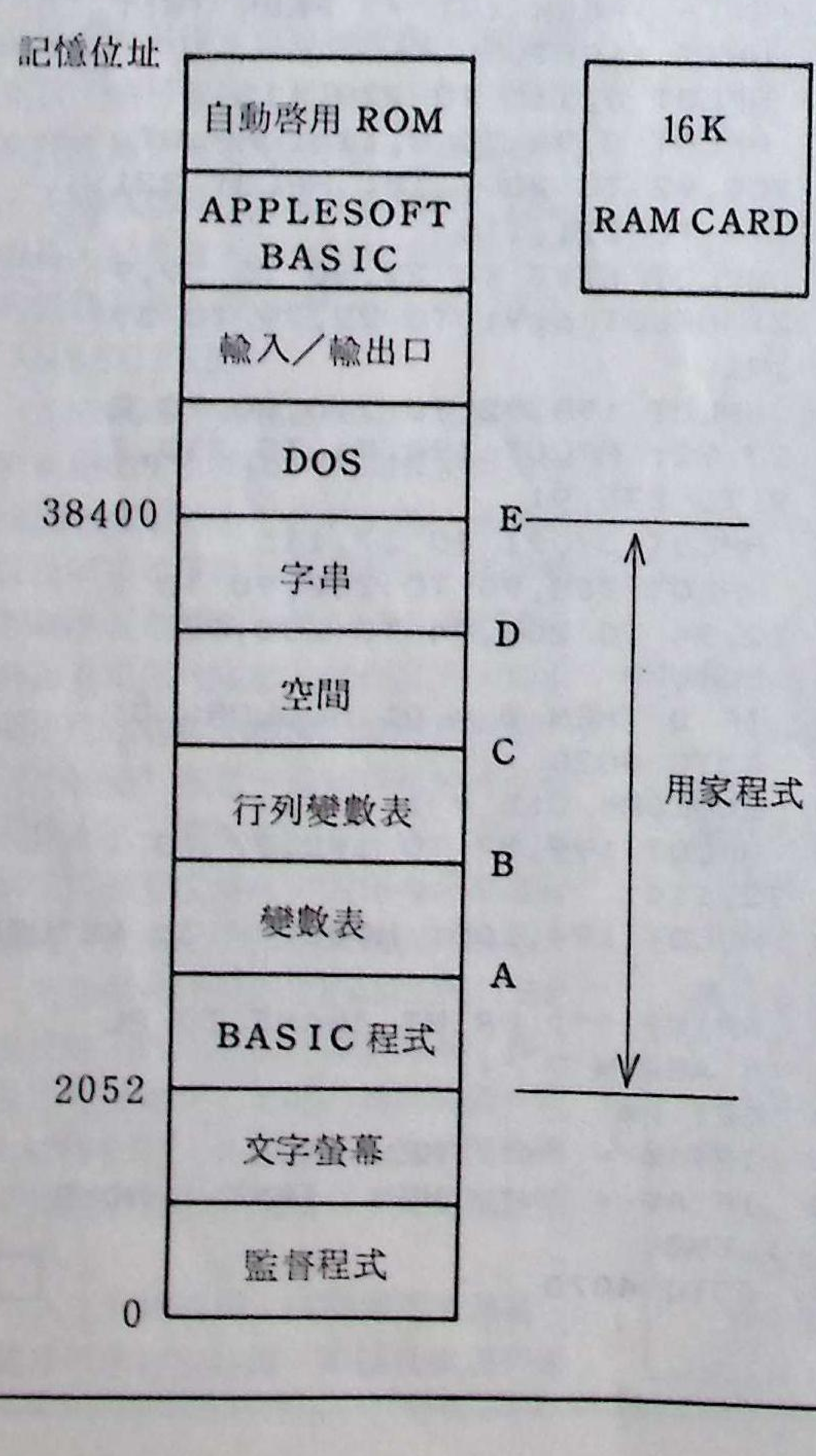
耐人研究的變數(上)

如何才能善用電腦的記憶空間，俾能在耗用大量記憶空間的程式如動畫，文字處理軟件等運作時能夠順利執行。從技巧上而言，.APPLESOFT 較易掌握，因此我們都樂意採用它，但程式的骨幹——變數，我們對它又知道多少呢？

在很多情況下，我們都會遇上一些語法，結構絕對沒有問題的程式，但在運行時，變數多了，便會破壞了高解像度圖像。或在開啓高解像圖形時破壞了變數表，亦破壞了程式。簡而言之，較長的 APPLESOFT 程式及其變數可能與高解像度的其中一頁圖形出現衝突，例如程式莫明奇妙的不見或由於某個變數不見了而無法繼續執行，便可肯定出現這種情況。

爲了清楚說明，以本刊第二期的〔數學套件程式〕爲例，它便有著以上的情形出現，因此必須將之載入高解像圖形所佔用的空間之上才能執行。這是衆多解決辦法其中的一個。不過怎樣才是最合適的解決辦法呢？我們可以研究程式的變數及位址分配，好能確保不發生以上情形。

48K的 APPLE 配置了磁碟機後，它的記憶空間便因 DOS 的佔用而減少了約 10.5 K。有關 APPLE Ⅱ 48 K 記憶空間分配如下圖所示：



它們的各個區間的位址是可以計算出來的：

例如A點是 APPLESOFT程式的最高點，亦是變數表的最低點。同時 LOMEM 的自設值亦相當於A點，它可以用 PEEK（105）＋256 ＊ PEEK（106）檢查出來。

B點則可以用 PEEK（107）＋256＊PEEK（108）計算出，它是變數表的頂點，行列變數表的最低點。

C點是行列變數表的最高點，亦是可用空間的起始點，可以用 PEEK（109）＋PEEK（110）＊256 檢查得到。由這開始直到D點是自由空間；基本上我們可以任意使用這些餘下空間，但閣下有沒有想到高解像圖形的第 1 和第 2 頁可能與A至C之間的位址相衝突呢？

至於D點是字串的存放最低位置，以 PEEK（111）＋256 ＊ PEEK（112）來計算出。

E點是 HIMEM 的自設值，是字串的最高存放位置，而且是 DOS 的位址最低點。

讓我們來看看到底變數是甚麼一回事：

一個變數是一個由英文單字及數字表示的代號（由A至Z的任何一個，再隨着有一個英文字母或任何零至九之間的數字），用來表示一個未知值，或是一個時常改變的值。使用變數可節省程式空間及減少執行時間，假設我們寫了 PR－INT 12245.87／4780 ，若我們需要重覆多次使用此表示式或參數，可先定A＝ 12245.87，B＝4780，再寫 PRINT A／B 便可以節省 10個 BYTE 記憶空間。而且程式也會執行得更快，因爲由 ASC Ⅱ 碼表示的 12245.87 及 4780 先前已在放入變數表時由 APPLESOFT 轉換成浮點數，因此而達到節省目的。

七種變數

APPLESOFT 比 INTEGER BASIC 有着更複雜的變數結構。 INTEGER BASIC 只有三種型態的變數，而且被存放在一個變數表中， APPLESOFT 有七種型態的變數及一種副型態變數，分別儲在三個不同憶體位址上。變數大體上可分兩類：基本變數及行列變數，基本變數又可分四種：實數、整數、字串及函數，前三種均有相同的行列變數，但函數則沒有。



耐人研究的變數(上)

APPLESOFT 藉着一個特殊的字元來區分變數的型別。一個實數沒有任何的識別號，整數則跟着一個百分號（%），字串跟着一個（$）號。但是這並不是直接編譯器內在的辨識法，APPLE 的語法分析常規在讀進每一個程式行時，將每一個變數放入變數表中，並附予一個不同組合的正負數。

變數的長度

任管 APPLESOFT 的每一個變數名稱可由238個字所組成，但實際上 APPLESOFT 只管前面兩個字，所以〝LEADPIPE〞及〝LEAPYEAR〞這兩個變數會被 APPLESOFT 視爲是同一個變數。同時 APPLESOFT 中的任一個指令或保留均不得爲變數的一部分。舉例說，一個變數的定義式＝ LENGTH＝1，會導致 APPLESOFT 產生〔? SYNTAX ERR〕，因爲此一變數 LENGTH 中包含了有保留字 LEN。當定義的變數名稱只有一個字時，APPLESOFT 會自動加一個虛（NULL）字元在名稱的第二個 BYTE 上。一個虛字的正值爲＄00（10進位數0），負值爲＄80（10進位數128）。

讀者應記在心裏，APPLESOFT 只認得任一個變數名稱的頭兩個字，但往往使用了不只兩個以幫助程式設計及標明每一變數之功能。當研讀程式時，舉例說 TABLE＝53248 總比X＝53248 能給人更佳的了解。

實數變數

在檢查出變數指標後，我們須清楚各種變數的性質，圖1中、將一個實數變數所佔用的七個 BYTE 予以列出。實數是僅有的實際佔用的七個 BYTE 的資料類別，所有簡單變數均佔用了七個 BYTE，但是 APPLESOFT 往往將一個變數的位址簡化，再放在變數表中。

前面兩個 BYTE 是正的 ASC Ⅱ，用來表示實數的名稱。第三個 BYTE 是指數部分接着的四個 BYTE 是數值部分，以高位 BYTE 起始，實數是幾種資料型別中最難懂的一種，有必要獨立研究此一資料型別。

例：A＝40000

BYTE	描述		16進位數	10進位數
0	VBLNAME CHR 1（POS）		41	65
1	VBLNAME CHR 2（POS）		00	0
2	指數		90	144
3	數值部分	高位 BYTE	1C	28
4			40	64
5			00	0
6		低位 BYTE	00	0

圖1、實數變數

註：VBLNAME 爲 VARIBLE NAME（變數名稱）

整數變數

整數在內部儲存時用負的 ASC Ⅱ來表示變數名稱，如圖2所示。在執行整數時的速度比實數還要慢，因爲在 APPLESOFT 中整數必須被轉換成浮點數才能處理。更由於一個整數名稱必須以一個百分號（%）來辨識，也佔用了記憶體額外的一個 BYTE，整數變數的使用最好限制在不希望使用浮點數的情況。

資料的儲放與其他地方有所不同，整數值的高位 BYTE 被放在前面，低位 BYTE 放在後面。這兩個 BYTE 緊跟在兩個代表名字的 BYTE 後，且其後接着三個沒有使用補以0的 BYTE。

例：B%＝32767

BYTE	描述		16進位數	10進位數
0	VBLNAME CHR 1（負）		C2	194
1	VBLNAME CHR 2（負）		80	128
2	高位 BYTE 值		7F	127
3	低位 BYTE 值		FF	255
4	未使用		00	0
5			00	0
6			00	0

圖2、整數變數

字串變數

字串變數用一對正負值的 ASCII來表示變數名稱，變數之內並不直接存在變數表中，而代之以在第四個及第五個 BYTE 存放資料所在之位址，參看圖3，第三個 BYTE 則存放字串的長度，第六個及第七個 BYTE 則未使用。

字串型別又可分兩種，且這兩種子型別，由於存在記憶體中的兩個不同區域，所以很容易混淆，這就是爲甚麼在變數表中只找到指至內容的指標的原因。第一種字串的子型別是在程式中直接定義一個字串的內容，如：100 A$＝"HELLO"，第二種則是由鍵盤或磁碟利用 INPUT 敍述作輸入的字串，如：110 INPUT A$。同時會被以與 HIMEM：相反的方向開始存放。

例：CD$＝"HELLO"

BYTE	描述	16進位數	10進位數
0	VBLNAME CHR 1（正）	43	67
1	VBLNAME CHR 2（負）	C4	196
2	字串長度	05	5
3	位置的低位 BYTE	FB	251
4	位置的高位 BYTE	8F	143
5	未使用	00	0
6		00	0

圖3、字串指標

函數變數

第四種類型的變數在 APPLESOFT 手冊上並無資料可尋，這是唯一沒有行例外觀的變數。當在程式中定義 DEF FN A(B)＝時，APPLESOFT 仍在變數表中建立了七個 BYTE 的變數指標，此七個 BYTE 之形式與內容可參看圖4，假若變數B並未被建立在變數表時，APPLESOFT 亦會幫B在變數表中函數變數指標後建立一個實數指標。

B這個變數是一個局部的（LOCAL）變數，若B在定義函數變數之前已加以定義，則 APPLESOFT 在使用函數時會先將B的值存在堆叠器中，等執行完函數後再予以重置（RESTORE），使堆叠器之值變爲零。

函數變數像其他類別的變數一樣，前面兩個 BYTE 是用來存放變數名稱，其第一個 BYTE 存放負的 ASC Ⅱ數值，第二個 BYTE 存放正的，第三個及第四個 BYTE 存放一個指標，指示在函數表示式或等號（＝）後面的定義式在記憶體的位置。接着的兩個 BYTE 指到函數變數執行後獲得值的變數所在記憶體之位置。第七個BYTE 存放局部的變數的第一個字，但其功能未知。

例：DEF FN EF (GH) ＝XY

BYTE	描 述	16進位數	10進位數
0	VBLNAME CHR 1 (負)	C5	197
1	VBLNAME CHR 2 (正)	46	70
2	定義式的低位 BYTE 位址	0E	14
3	定義式的高位 BYTE 位址	08	8
4	GH 的低位 BYTE 位址	1B	27
5	GH 的高位 BYTE 位址	08	8
6	正的 ASC Ⅱ 表示＝後的字	58	88

圖4、函數指標

執筆至此，相信讀者都會對變數有點概念了，在下一期，筆者將會舉出一個程式幫助讀者尋找任何 APPLESOFT 程式中的變數及其所在位置。□

第7至8期程式磁碟總目錄

7

(1) 會說話的蘋果
(2) 私人財政預算系統
(3) 私人收入會計系統
(4) 高解像字符繪劃器
(5) 圖像轉載器
(6) 螢幕文本編輯處理器
(7) 濃縮圖片程式修訂加強版本
(8) 蘋果杯大賽車
(9) 十六個難度的食鬼遊戲
(10) 高爾夫球遊戲。
(11) 空戰驚魂
(12) 美麗圖案設計家
(13) 更多色彩
(14) 列印的困擾
(15) 十一條空 SECTOR
(16) 捉蟲訓練
(17) 80字行咭例子程式

8

(一) APPLESOFT滙編器
(二) 最簡單的拆帶程式
(三) 英文打字練習
(四) 編印單個LABEL
(五) 高解像頁處理工具箱
(六) 簡易快速的記憶移動方法
(七) 眞正要動腦筋的小學四則運算
(八) 七海霸王 (DEPTH CHARGE)
(九) 原子黑洞
(十) 蛇之噩夢
(十一) 收銀機
(十二) 改良的閃動訊息
(十三) 兩欄索引顯示
(十四) 自動三角函數曲綫繪劃系統
(十五) 任意鍵盤 (三個)
(十六) 每期捉蟲問題
(十七) 奇妙的 &

第7,8 期程式磁帶

編號：C#7

(一) 十六種難度的食鬼遊戲
(二) 更多色彩
(三) 空戰驚魂
(四) 捉虫問題
(五) 捉虫答案

編號：C#8

(一) 最簡單的拆帶程式
(二) 英文打字練習
(三) 原子黑洞
(四) 蛇之噩夢
(五) 收銀機
(六) 任意鍵盤

PRODOS縱橫

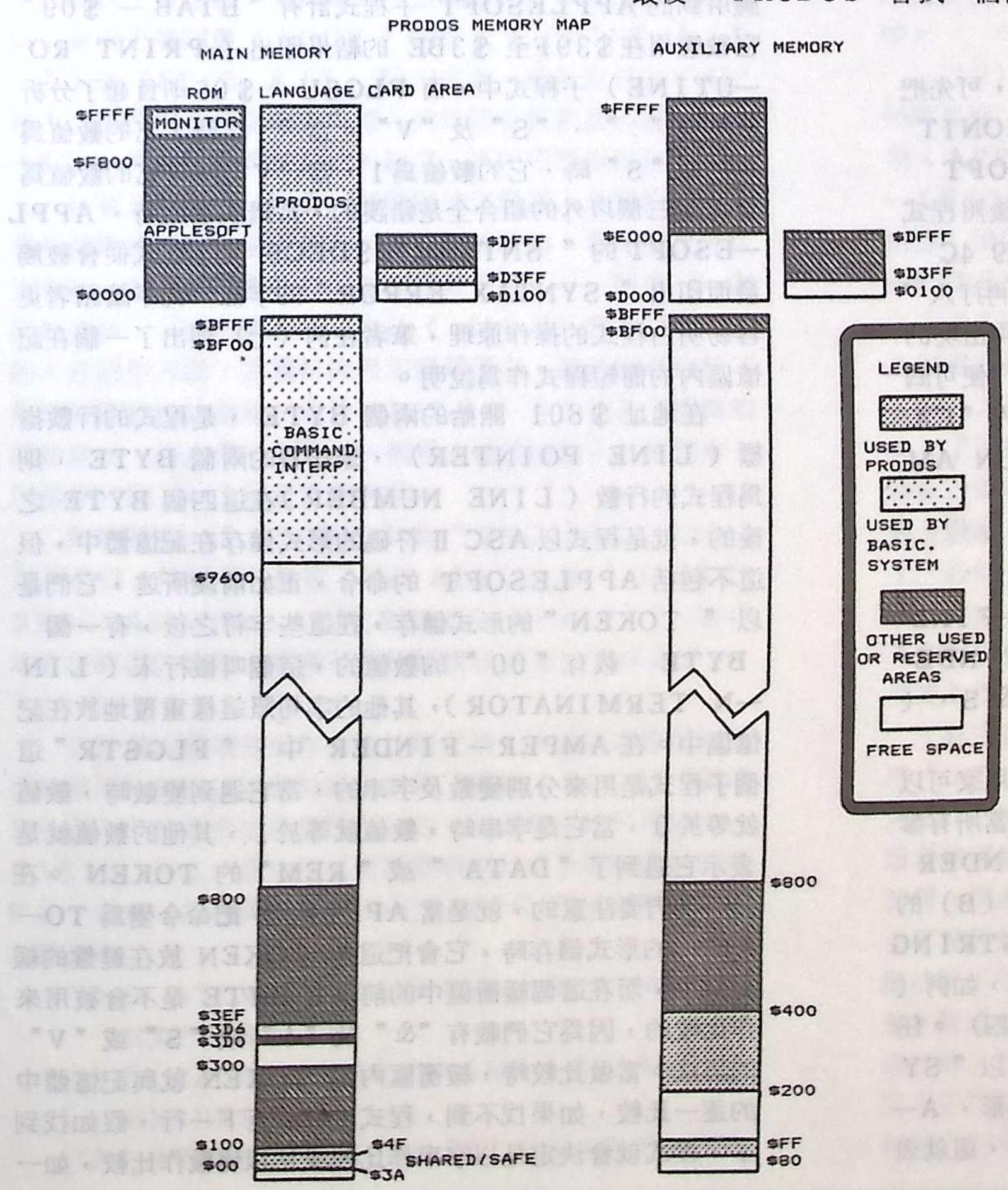
普魯斯

第3篇：啟動過程及記憶體之使用技術

當我們啓動一張PRODOS磁碟時，在CONTROLLER CARD的RO—M控制程式將磁碟上第0及1兩個BLOCK上的代碼讀入，存放到記憶體$800位置並執行之，這時分別出使用者的電腦是APPLEⅡ或APPL—EⅢ，分別將PRODOS或SOS KERNEL兩個檔案之中一個載入。因此PRODOS能在任何型號上啓動，在此不討論APPLEⅢ，而將焦點放在APPLEⅡ上。

PRODOS檔案載入到$2000位址後便會被自動執行，使PRODOS系統存放到BANK—SWITCH記憶體（BSM）上的$D000—$DFFF位置，而將某機械語言界面（MACHI—NE LANGUAGE INTERFAC—E），簡稱MLI，存在$E000至$FFFF上，作爲磁碟上檔案及傳譯工具，之後便在$BF00至$BFFF設立一頁256個BYTE的共通記憶區（SYSTEM GLOBAL PAGE），使PRODOS能搜索首個傳譯檔案，即磁碟上的檔案以〝SYSTEM〞作爲結尾語，通常此檔案會是BASIC傳譯程式（BI），名爲BASIC．SY—STEM的檔案。它被載入到$2000的區域後再被移至$9A00至$BDFF，並將HIMEM設立在$9600。

最後，PRODOS會找一個檔案叫STARTUP的系統程式，（它可以是SYS系統程式，BIN二進制程式TXT文字程式，BAS BASIC程式的檔案），當載入後，STARTUP會被執行，這樣便結束了PRODOS的啓動過程。



記憶體的使用

除了上述的記憶體使用之外，第零頁即$00至$FF之內$3A至$4F亦被PRODOS的機械碼翻譯器MLI所佔用。$D100至$DFFF被保留作爲以後用途；$80至$FF，$200至$3FF，$BF00至$BFFF，$E000至$FFFF，APPLE Ⅱe的EXTENDED 80 COLUMN CARD上的D100至$DFFF亦保留作以後之用。

RAMDISK模擬

當PRODOS在一部擁有128K的APPLEⅡe上啓動時，另外的64K會連通至SLOT 3，DRIVE作爲模擬磁碟機之用，它的VOLUME PATHNAME是／RAM，這個RA—M DISK有著12個空置的BLOCK可供存放資料，而它的VOLUME D—IRECTORY可存入不超過12個檔案名字。

RAM DISK的存取速度是很快的，不過若這64K RAM被其它的程式佔用時，RAM DISK內的資料便會被破壞，所以，它在這方面比128K RAM咭遜色，因爲128K RAM咭模擬爲磁碟機時，在內的資料被保護，就算重新BOOT機（只要不關掉電源）亦不破壞其內的資料。

有關這方面將來再加以討論，下期將探討SYSTEM GLOBAL PAGE及其他記憶體的使用情形，相信以上及以後的資料對PRODOS的程式設計會很有幫助，起碼在各位未擁有TEC—HNICAL MANUAL之前，尤應注意，再會！□

THOMAS

&專欄

快速尋找及磁碟矩陣儲存與提取

在上次的文章中與各位讀者討論了"&"的邏輯特性及應用實例，今次再提供多兩個應用程式，供各位在處理文字尋找、資料在磁碟間之儲存及提取方面，一個更方便快捷的方法。

(1)快速尋找

AMPER—FINDER—用 6502 機械語言編滙而成的程式，它能夠地極短的時間內找到了程式中的變數，字串，命令及隱蔽的字母（CONTROL CHARACTER）等。程式的設計，是放在 APPLE Ⅱ 的第三頁，意即由 $300開始，而且可以在任何時間載入電腦中而不影响主程式的操作。除了這些簡單的用途之外，用家還可以利用這個程式，去找出主程式的"GOTO"及"GOSUB"命令，又或找尋某些變數，以確定其最初之數值及執行後之數值，此舉能令到用家在程式除錯方面得到莫大的幫助，又或找出程式中打錯了的字等等。

程式敘述

本程式是磁碟版本，用家中如有使用錄音機的，可先把此程式打進 APPLE Ⅱ 的 6502 監督系統（MONIT—OR）中，做法是當 APPLE Ⅱ 在 APPLESOFT BASIC 的環境中打入"CALL—151"，然後照程式的數據逐一打入地址 $ 300 中，即"＊300：A9 4C 8D F503"，當全部輸入妥當後，預備好錄音機，再打入"300.3C1W"，這時按下紅色的錄音掣，待游標再出現的時候，程式已儲存在錄音帶中了，按"CTRL—C"便可回到 APPLESOFT 中，要執行這個程式，只需打入"CALL 768"便可，有磁碟機的讀者只需打入"BRUN AM—PER—FINDER"則可。

程式語法

當程式被觸發後，用家便可利用"AMPER—FIND—ER"的語法，作出種種的找尋。AMPER—FINDE—R 的語法很簡單，分別爲—（A）&：，（B）&S，（C）&V。

（A）的語法爲通用的格式，在"："之後，用家可以給予任何的變數，或字串，或命令，或隱蔽字母，當所有參數輸入後，按〔RETURN〕，則 AMPER—FINDER 找到後便會把相關的字行號數顯示，如例（二）。（B）的例子則比較專門一些，"S"的意思就是字串（STRING），即是這個語法是用來尋找主程式的字串或字母，如例（三）。（C）的語法是尋找主程式的變數，如例（四）。任何不是上述三種規定語法的，APPLE Ⅱ 都會報以"SYNTAX ERROR"，如例（五）。除却以上的情形，A—MPER—FINDER 會執行完而不給任何顯示的，這就表示了有以下的兩種可能性：

（一）AMPER—FINDER 找不到用家提供的參數。

（二）AMPER—FINDER 找到了參數包含了 APPLESOFT 的"TOKEN"，如例（六）。如"&：LETTER"及"& SAVENUE"的語法，AM—PER—FINDER在前者被 TOKEN—"LET"誤導了而找不到，而後者的語法又使 AMPER—FINDER 誤會了是"SAVE"的"TOKEN 而顯示了"SYNTAX ERROR"的報導，這點應是用家所要留意的。

程式的結構

與一般的"&"程式一樣，在地址 $ 300—$30F一段，作爲程式的觸發點，跳至"&"的向量地址 $3F5 中，而 $3F5 則爲本程式的起始點。爲了盡量縮短程式的長度，它使用了 APPLESOFT 的零頁區作爲指標及疊存區，其所應用到的 APPLESOFT 子程式計有"HTAB— $09"它被應用在 $39F至 $3BE 的結果印出（PRINT RO—UTINE）子程式中，而 FLGSU— $06則負起了分析語法中"："，"S"及"V"。當"："時，它的數值爲0，當"S"時，它的數值爲1，當"V"時，它的數值爲2，這三個以外的組合全是錯誤的，當錯誤發生時，APPL—ESOFT 的"SNTXER—$DEC9"子程式便會被觸發而印出"SYNTAX ERROR"的字樣。爲了讓讀者更容易明白程式的操作原理，筆者在例（七）列出了一個在記憶區內的簡短程式作爲說明。

在地址 $801 開始的兩個 BYTE，是程式的行數指標（LINE POINTER），緊接着的兩個 BYTE，則爲程式的行數（LINE NUMBER）在這四個 BYTE 之後的，就是程式以 ASC Ⅱ 符碼的形式儲存在記憶體中，但這不包括 APPLESOFT 的命令，正如前段所述，它們是以"TOKEN"的形式儲存，在這些字符之後，有一個 BYTE 載有"00"的數值的，這個叫做行末（LIN—N TERMINATOR），其他的字句照這樣重覆地放在記憶區中。在 AMPER—FINDER 中，"FLGSTR"這個子程式是用來分別變數及字串的，當它遇到變數時，數值就等於0，當它是字串時，數值就等於1，其他的數值就是表示它遇到了"DATA"或"REM"的 TOKEN。在這裡我們要注意的，就是當 APPLE Ⅱ 把命令變爲 TO—KEN 的形式儲存時，它會把這個 TOKEN 放在鍵盤的緩衝區中，而在這個緩衝區中的前兩個 BYTE 是不會被用來作比較的，因爲它們載有"&"及"："或"S"或"V"的語法。當做比較時，緩衝區內的 TOKEN 就與記憶體中的逐一比較，如果找不到，程式就會跳至下一行，假如找到了，程式就會決定是以字串作比較或是以變數作比較，如一

切順利，程式便會把控制權交到〝PRINT〞的子程式中。

〝PRINT〞子程式牽涉了兩個 APPLESOFT 的程式，它們是 $ED24 的 LINPRT，負責印出行數，$DAFB 的 CRDO，負責印出 CR 的訊號，在每一次的過程中，它都印出相應的字行數目，而 HTAB 則累加6，直至36為止，CR 的訊號便印出，令到印字機回到下一行中，HTAB 便重置回 0。這個次序不停地重覆著直至行數指標的數值為0，然後經由 $D43C 跳回 APPLESOFT 中，這解釋了程式的邏輯流程。各位讀者如有興趣研究這程式便不難明白程式的特性。

(2)矩陣儲存與提取

在 APPLESOFT 中有兩個命令是用來處理大量的矩陣資料，但這兩個命令，却鮮有用家使用，這裡要提及的，就是〝STORE〞及〝RECALL〞。鮮為人用的原因，據筆者分析，不離兩個原因，第一就是這個命令主要用來將矩陣資料儲存在磁碟中，而現在大多數的微電腦用家都已擁有磁碟機，故此把它冷落在一旁。第二個原因就是它的儲存及提取速度比較慢，舉個例說，要儲存一個50×50共 2,601 個資料，時間就耗去了大約 270 秒。讀者可能認為利用磁碟會比較快，但這想法却是錯的，要儲存或提取同樣份量的資料，筆者用雙回環（LOOP）的程式〔PRINT A（I，L）及 INPUT A（I，J）〕去做，結果就耗去了412秒！這是因為任何從磁碟中讀入的矩陣資料都要經過一個由 ASC II 符碼轉換到16進位的數字，再把它們儲錄在 RAM 上。在整個過程中，每當一個扇區的資料讀入微電腦時，都要做轉換的工作，磁碟便在停止等候的狀態；當轉換及儲存完畢後，磁碟機又再轉動，重覆地做著讀入——轉換——儲存的步驟，其中所耗去時間之長（412 秒）實是可以理解的。在磁帶方面，其邏輯流程却簡單得多，這是因為資料不需要經過轉換的過程，而是 APPLE II 首先找著了矩陣的開始地址，接著便一個緊接一個地由磁帶直讀入記憶體中，這樣便省回了142 秒寶貴的時間。

如前所述，〝STORE〞及〝RECALL〞這個命令並不是每一位讀者都熟識，但在 APPLE 的〝BASIC PROGRAMMING REFERENCE MANUAL〞一書中却有很詳盡的論述，讀者可先行閱讀以便更能了解本文。。

由於使用磁碟的儲存方法比較便捷，故此筆者在這裡提供的程式是利用了這個特點，運用了它的〝BLOAD〞及〝BSAVE〞的命令去達致這個目的。而這個程式又可以由 BASIC 程式中自由呼叫而不影响主程式及矩陣資料的完整性。程式（見程式二）可以由 BASIC 程式相呼叫，而其語法有二：

（1）& STORE ARRAYNAME，STRING EXPRESSION

（2）& RECALL ARRAYNAME，STRING EXPRESSION

〝&〞令程式觸發了 $3F5的〝&〞向量，然後，程式便會找尋〝STORE〞及〝RECALL〞的 TOKEN，如果沒有找到，APPLE 就會報以〝SYNTAX ERROR〞的句子，如果找到了，程式就會繼續找矩陣的名字。因為本程式是利用碟帶的儲存及提取的特點，所以用家給予的矩陣名稱一定要預早有所設定（DEFINE），否則 APPLE 便會報以〝OUT OF DATA〞或〝TYPE MISMATCH ERROR〞的句子。假設一切順利，第一個矩陣資料單元便由一個指標作紀錄，相應地這個指標及整個矩陣的長度便會加以計算並紀錄。再者，程式會找尋逗號（COMMA)的位置而分析字串的結構，如是者經以上所分析過之資料，再經程式的組合後使之變為一個 DOS 可以接受的檔案名稱，將整個矩陣用〝BSAVE FILENAME，A$×××，L$××××〞或〝BLOAD FILENAME，A$××××〞的形式儲存在磁碟中。

當〝STORE〞的命令執行完畢之後，控制權便迅速回到 APPLESOFT 中，但如果矩陣是由磁碟中讀入 APPLE 的記憶體後，程式就會根據〝RECALL〞所說明的矩陣資料長度與磁碟輸入的檔案長度作一比較，這便產生了三個不同的結果：

（1）假如在〝RECALL〞中說明的長度與輸入的矩陣長度相等，則一切沒有問題，控制權交回APPLESOFT 中。

（2）假如在〝RECALL〞中說明的長度大於輸入的矩陣長度，問題亦不嚴重，但 APPLE 會發出警告性的一响，APPLE 報以〝ERR〞的句子，控制權交回 APPLESOFT 中，主程式繼續操作。這個情形與磁帶的提取相似，讀者可翻閱〝BASIC PROGRAMMING REFERENCE MANUAL〞一書內之第62至64頁。

（3）假如在〝RECALL〞中說明的矩陣長度小於輸入檔案之長度，這便引起了嚴重的惡性後果，在此情況底下，輸入之檔案便會因此而崩潰（CORRUPT），使到資料失却了連續性及完整性，為了防止這個情況發生，程式會即時執行清除（CLEAR)的命令，以防止不可挽救的程式崩潰及毀滅，接着 APPLE 會報以〝OUT OF MEMORY〞的句子，控制權交回 APPLESOFT 中。在這裡得向讀者提出一個警告，因為矩陣的長度是要等待整個檔案的輸入才可以計算及加以比較，假如矩陣的長度大至佔據了本程式或 DOS 的佔用區時，後果便慘不忍睹了。

程式在地址 $8400 開始，要執行本程式，讀者可以利用〝APPLESOFT TOOL KIT〞的 ASSEMBLER／EDITOR 先行把程式組合，然後〝BRUN〞便可，但讀者亦可把程式直接輸入 APPLE 的記憶體中，方法已述於前，供各位讀者參考。程式在處理一個50×50的矩陣只耗去了25秒的時間，相對來說，它比原來 APPLE 所提供的辦法快了差不多16倍！而磁碟的儲存空間也比原來的小了50%。程式也可以在 APPLE 執行 BASIC 程式中遇到錯誤時，把矩陣的資料儲存，當 BASIC 的錯誤修正後，又可以把矩陣資料還原，使到程式在執行時不用耗費大量的時間作計算，另外在除錯（DEBUG)時提供莫大的幫助。

快速尋找及磁碟矩陣儲存與提取

例（一）

```
10  HOME
20  PRINT "THIS IS AN EXAMPLE"
30  PRINT "STRING - 12345 67890"
40 K = 12345
50 KK = 67890
60  END
```

例（二）

```
]&:
6       10      20      30      40      50
60
]
```

例子7

```
]FUNCTION? H/A DUMP $ 800

0800- 00 07 08 0A 00 97 00 21  .......!
0808- 08 14 00 BA 22 54 48 49  ...:"THI
0810- 53 20 49 53 20 41 4E 20  S IS AN
0818- 45 58 41 4D 50 4C 45 22  EXAMPLE"
0820- 00 3D 08 1E 00 BA 22 53  .=...:"S
0828- 54 52 49 4E 47 20 2D 20  TRING -
0830- 31 32 33 34 35 20 36 37  12345 67
0838- 38 39 30 22 00 47 08 28  890".I.(
0840- 00 4B D0 31 32 33 34 35  .KP12345
0848- 00 56 08 32 00 4B 4B D0  .V.2.KKP
0850- 36 37 38 39 30 00 5C 08  67890.\.
0858- 3C 00 80 00 00 00 00 00  <.......
0860- 00 00 00 00 00 00 00 00  ........
0868- 00 00 00 00 00 00 00 00  ........
```

程式一

```
SOURCE FILE: AMPER FINDER
0000:              *
0000:              * AMPER-FINDER
0000:              *
0000:              * (C) MICRO ORCHARD 84
0000:              *
0000:              * DISTRIBUTED BY COMPUHOUSE
0000:              *
----- NEXT OBJECT FILE NAME IS AMPER FINDER.OBJO
0300:                        ORG  $300
0007:                FLGSTR  EQU  $07
0006:                FLGSV   EQU  $06
0008:                YSTO    EQU  $08
0009:                HTAB    EQU  09
0018:                LSTART  EQU  $18
001A:                LINENO  EQU  $1A
001C:                FNEXT   EQU  $1C
0024:                HPOS    EQU  $24
0200:                KEYBRD  EQU  $200
03F5:                VECTOR  EQU  $3F5
D43C:                BASIC   EQU  $D43C
DAFB:                CRDO    EQU  $DAFB
DEC9:                SNTXER  EQU  $DEC9
ED24:                LINPRT  EQU  $ED24
0300:              *
0300:              * SET &-JUMP VECTOR
0300:              *
0300:A9 4C                   LDA  #$4C
0302:8D F5 03                STA  VECTOR
0305:A9 10                   LDA  #START
0307:8D F6 03                STA  VECTOR+1
030A:A9 03                   LDA  #$03
030C:8D F7 03                STA  VECTOR+2
030F:60                      RTS
0310:              *
0310:              * SET POINTERS
0310:              *
0310:A0 00           START   LDY  #$00      ZERO
0312:84 09                   STY  HTAB      HTAB POINTER AND
0314:84 06                   STY  FLGSV     STR-VAR FLAG
0316:              *
0316:C8                      INY            SET FNEXT
0317:84 1C                   STY  FNEXT     TO STARTING
0319:A9 08                   LDA  #$08      ADDRESS OF
031B:85 1D                   STA  FNEXT+1   PROGRAM
031D:              *
031D:AD 01 02                LDA  KEYBRD+1  COMMAND?
0320:C9 3A                   CMP  #$3A      ASCII ':'?
0322:F0 10                   BEQ  NXTLN     YES!
0324:84 06                   STY  FLGSV     SET STR-VAR FLAG=1
0326:C9 53                   CMP  #$53      ASCII 'S'?
0328:F0 0A                   BEQ  NXTLN     YES!
032A:C8                      INY
032B:84 06                   STY  FLGSV     SET SR-VAR FLAG=2
032D:C9 56                   CMP  #$56      ASCII 'V'?
032F:F0 03                   BEQ  NXTLN     YES!
0331:4C C9 DE                JMP  SNTXER    SYNTAX ERROR
0334:              *
0334:A5 1C           NXTLN   LDA  FNEXT     SET LINENO
0336:18                      CLC             TO ADDRESS
0337:69 02                   ADC  #$02      OF CURRENT
0339:85 1A                   STA  LINENO    LINE NUMBER
033B:A5 1D                   LDA  FNEXT+1
033D:69 00                   ADC  #$00
033F:85 1B                   STA  LINENO+1
0341:              *
0341:A5 1A                   LDA  LINENO    SET LSTART
0343:18                      CLC             TO ADDRESS
0344:69 02                   ADC  #$02      OF CURRENT
0346:85 18                   STA  LSTART    LINE START
0348:A5 1B                   LDA  LINENO+1
034A:69 00                   ADC  #$00
034C:85 19                   STA  LSTART+1
034E:              *
034E:A0 01                   LDY  #$01      SET FNEXT
0350:B1 1C                   LDA  (FNEXT),Y TO NEXT
0352:F0 6B                   BEQ  END       LINE POINTER
0354:AA                      TAX
0355:88                      DEY
0356:B1 1C                   LDA  (FNEXT),Y
0358:85 1C                   STA  FNEXT
035A:86 1D                   STX  FNEXT+1
035C:              *
035C:              * MAIN SEARCH ROUTINE
035C:              *
035C:84 07                   STY  FLGSTR
035E:88                      DEY
035F:A2 FF           STEP3   LDX  #$FF
0361:C8              STEP4   INY
0362:E8                      INX
0363:D0 02                   BNE  STEP5
0365:84 08                   STY  YSTO
0367:BD 02 02        STEP5   LDA  KEYBRD+2,X
036A:C9 00                   CMP  #$00
036C:D0 0F                   BNE  STEP6
036E:A5 06                   LDA  FLGSV     PRINT ALL?
0370:F0 2D                   BEQ  FPRINT    YES!
0372:38                      SEC
0373:E5 07                   SBC  FLGSTR    PRINT STRING?
0375:F0 28                   BEQ  FPRINT    YES!
0377:C9 02                   CMP  #$02      PRINT VARIABLE?
0379:F0 24                   BEQ  FPRINT    YES!
037B:D0 1D                   BNE  STEP8
037D:B1 18           STEP6   LDA  (LSTART),Y
037F:F0 B3                   BEQ  NXTLN     NEXT LINE
0381:C9 22                   CMP  #$22       " TOKEN
0383:F0 08                   BEQ  SWFLG
0385:C9 83                   CMP  #$83
0387:F0 04                   BEQ  SWFLG
0389:C9 82                   CMP  #$82
038B:D0 06                   BNE  STEP7     REM TOKEN
038D:A5 07           SWFLG   LDA  FLGSTR
038F:49 01                   EOR  #$01
0391:85 07                   STA  FLGSTR
0393:B1 18           STEP7   LDA  (LSTART),Y
0395:DD 02 02                CMP  KEYBRD+2,X
0398:F0 C7                   BEQ  STEP4
039A:A4 08           STEP8   LDY  YSTO
039C:4C 5F 03                JMP  STEP3
039F:              *
039F:              * PRINT ROUTINE
039F:              *
039F:A0 00           FPRINT  LDY  #$00      PRINT LINE
03A1:B1 1A                   LDA  (LINENO),Y NUMBER
03A3:AA                      TAX
03A4:C8                      INY
03A5:B1 1A                   LDA  (LINENO),Y
03A7:20 24 ED                JSR  LINPRT
03AA:A5 09                   LDA  HTAB      HTAB TO
03AC:18                      CLC             NEXT FIELD
03AD:69 06                   ADC  #$06
03AF:C9 24                   CMP  #$24      TIME FOR CR?
03B1:D0 05                   BNE  STEP9
03B3:20 FB DA                JSR  CRDO      YES!
03B6:A9 00                   LDA  #$00
03B8:85 09           STEP9   STA  HTAB
03BA:85 24                   STA  HPOS
03BC:4C 34 03                JMP  NXTLN
03BF:              *
03BF:              * RETURN TO APPLESOFT
03BF:              *
03BF:4C 3C D4        END     JMP  BASIC

*** SUCCESSFUL ASSEMBLY: NO ERRORS
```

```
D43C BASIC       DAFB CRDO       03BF END         07 FLGSTR
  06 FLGSV         1C FNEXT      039F FPRINT        24 HPOS
  09 HTAB        0200 KEYBRD       1A LINENO      ED24 LINPRT
  18 LSTART      0334 NXTLN      DEC9 SNTXER      0310 START
035F STEP3       0361 STEP4      0367 STEP5       037D STEP6
0393 STEP7       039A STEP8      03B8 STEP9       038D SWFLG
03F5 VECTOR        08 YSTO
```

```
  06 FLGSV         07 FLGSTR       08 YSTO          09 HTAB
  18 LSTART        1A LINENO       1C FNEXT         24 HPOS
0200 KEYBRD      0310 START      0334 NXTLN       035F STEP3
0361 STEP4       0367 STEP5      037D STEP6       038D SWFLG
0393 STEP7       039A STEP8      039F FPRINT      03B8 STEP9
03BF END         03F5 VECTOR     D43C BASIC       DAFB CRDO
DEC9 SNTXER      ED24 LINPRT
```

快速尋找及磁碟矩陣儲存與提取

例（三）	例（四）	例（五）	例（六）
]&S12345	]&V12345	]&12345	]&:HOME
30	40	? SYNTAX ERROR	10
]			

程式二

```
SOURCE FILE: AMPER DISK STORE AND RECALL
0000:              *
0000:              * & STORE, & RECALL
0000:              *
0000:              * (C) MICRO ORCHARD 1984
0000:              *
0000:              * DISTRIBUTED BY COMPUHOUSE
0000:              *
----- NEXT OBJECT FILE NAME IS AMPER DISK STORE AND RECALO
8400:                      ORG  $8400
8400:              *
8400:              * INIT & JMP AND SET HIMEM
8400:              *
0073:              HIM     EQU  $73        HIMEM
8400:              *
8400:A9 00         START   LDA  #START
8402:85 73                 STA  HIM
8404:A9 84                 LDA  #$84
8406:85 74                 STA  HIM+1      INIT HIMEM
8408:A9 13                 LDA  #ENTS
840A:8D F6 03              STA  BJP+1
840D:A9 84                 LDA  #$84
840F:8D F7 03              STA  BJP+2      INIT & VECTOR
8412:60                    RTS             RETURN
8413:              *
8413:              * DATA
8413:              *
0011:              STTS    EQU  $11
00B1:              CHNW    EQU  $B1        CHAR GET ROUTINE
0076:              RUNB    EQU  $76        RUN FLAG
0084:              PSTA    EQU  $84        POINTER TO FIRST ELEMENT
009B:              SARA    EQU  $9B        POINTER TO MATRIX
00FF:              MSK     EQU  $FF        STORE/RECALL FLAG
03F5:              BJP     EQU  $3F5       &JUMP VECTOR
0A60:              DS      EQU  $A60       LENGTH BLOADED FILE
F7D9:              GNAME   EQU  $F7D9      FIND MATRIX NAME
FF2D:              PRERR   EQU  $FF2D      PRINT 'ERR'
D66C:              CLEAR   EQU  $D66C      CLEAR BASIC VARIABLES
F7C9:              CALL    EQU  $F7C9      CALC POINTER TO FIRST EL.
DEBE:              CKOM    EQU  $DEBE      CHECK ON COMMA
DD7B:              EEXP    EQU  $DD7B      EVALUATE EXPRESSION
E600:              DR2     EQU  $E600      PREPARE PRINT
DB40:              DR3     EQU  $DB40      PRINT FILE NAME
D43C:              WBSIC   EQU  $D43C      WARM START BASIC
FDDA:              OHEX    EQU  $FDDA      PRINT 2 HEX DIGITS
DB5C:              COUT    EQU  $DB5C      CHAR OUT ROUTINE
DAFB:              CROUT   EQU  $DAFB      CARRIAGE RETURN
DEC9:              SYN     EQU  $DEC9      SYNTAX ERROR
D410:              OMEM    EQU  $D410      OUT OF MEMORY ERROR
DD76:              TYPM    EQU  $DD76      TYPE MISMATCH ERROR
8413:              *
8413:              * MAIN PROGRAM
8413:              *
8413:C9 A7         ENTS    CMP  #$A7       IS IT RECALL ?
8415:F0 07                 BEQ  YST        BRANCH IF SO
8417:E9 A8                 SBC  #$A8       IS IT STORE ?
8419:F0 03                 BEQ  YST        BRANCH IF SO
841B:4C C9 DE              JMP  SYN        SYNTAX ERROR IF NOT
841E:85 FF         YST     STA  MSK        SAVE STORE/RECALL FLAG
8420:20 B1 00              JSR  CHNW       GET NEXT CHAR
8423:20 D9 F7              JSR  GNAME      FIND NAME MATRIX
8426:24 11                 BIT  STTS       CHECK IF STRING ARRAY
8428:10 03                 BPL  NSTR       BRANCH IF NOT
842A:4C 76 DD      TPER    JMP  TYPM       TYPE MISMATCH ERROR
842D:20 C9 F7      NSTR    JSR  CALL       CALCULATE POINTER TO
8430:18                    CLC              FIRST ELEMENT OF MATRIX

8431:A0 02                 LDY  #$02
8433:B1 9B                 LDA  (SARA),Y
8435:65 9B                 ADC  SARA       CALCULATE LENGTH
8437:AA                    TAX              (IN MEMORY) OF
8438:C8                    INY              MATRIX
8439:B1 9B                 LDA  (SARA),Y
843B:65 9C                 ADC  SARA+1
843D:A8                    TAY
843E:8A                    TXA
843F:38                    SEC
8440:E5 84                 SBC  PSTA
8442:48                    PHA              PUSH LENGTH ON STACK
8443:98                    TYA
8444:E5 85                 SBC  PSTA+1
8446:48                    PHA
8447:20 BE DE              JSR  CKOM       CHECK ON COMMA
844A:20 7B DD              JSR  EEXP       EVALUATE EXPRESSION
844D:24 11                 BIT  STTS       CHECK IF STRING
844F:10 D9                 BPL  TPER       ERROR IF NOT
8451:20 00 E6              JSR  DR2        PREPARE PRINT
8454:48                    PHA
8455:A4 FF                 LDY  MSK        SET FLAG
8457:F0 02                 BEQ  PBLD       BRANCH IF STORE
8459:A0 06                 LDY  #$06       LOAD MESSAGE POINTER
845B:20 AB 84      PBLD    JSR  PMES       OUTPUT BSAVE/BLOAD
845E:68                    PLA
845F:20 40 DB              JSR  DR3        OUTPUT FILE NAME
8462:A0 0C                 LDY  #$0C
8464:20 AB 84              JSR  PMES       OUTPUT ",A$" TO DOS
8467:A5 85                 LDA  PSTA+1
8469:20 DA FD              JSR  OHEX
846C:A5 84                 LDA  PSTA       MATRIX TO DOS
846E:20 DA FD              JSR  OHEX
8471:A5 FF                 LDA  MSK        GET FLAG
8473:D0 16                 BNE  CEK        BRACH IF RECALL
8475:20 B5 84              JSR  REP        OUTPUT ",L$" TO DOS
8478:68                    PLA
8479:20 DA FD              JSR  OHEX       OUTPUT LENGTH MATRIX
847C:68                    PLA              TO DOS
847D:20 DA FD              JSR  OHEX
8480:20 FB DA              JSR  CROUT      BSAVE MATRIX
8483:24 76         CHRN    BIT  RUNB       IMMED EXECUTION MODE ?
8485:10 03                 BPL  RT1        BRACH IF NOT
8487:4C 3C D4              JMP  WBSIC      ELSE WARM START TO BASIC
848A:60            RT1     RTS              RETURN TO BASIC
848B:20 FB DA      CEK     JSR  CROUT      BLOAD MATERIX
848E:68                    PLA              CHECK WHETHER IT FITS
848F:CD 61 0A              CMP  DS+1
8492:90 11                 BCC  OM         ERROR, FILE TOO LARGE
8494:F0 07                 BEQ  IEQ        MAYBE OK
8496:68                    PLA
8497:20 2D FF      TSM     JSR  PRERR      ERROR, FILE TOO SMALL
849A:4C 83 84              JMP  CHRN       BUT CONTINUE EXECUTION
849D:68            IEQ     PLA
849E:CD 60 0A              CMP  DS
84A1:F0 E0                 BEQ  CHRN       FILE FITS EXACTLY
84A3:B0 F2                 BCS  TSM        FILE TOO SMALL
84A5:20 6C D6      OM      JSR  CLEAR      FILE TOO LARGE, CLEAR
84A8:4C 10 D4              JMP  OMEM       VARIABLES AND PRINT ERROR
84AB:              *
84AB:              * SUBROUTINE PRINT
84AB:              *
84AB:B9 B8 84      PMES    LDA  TXT,Y
84AE:08                    PHP
84AF:20 5C DB              JSR  COUT       PRINT CHARACTER
84B2:28                    PLP              FIRST BIT ZERO ?
84B3:10 D5                 BPL  RT1        READY IF SO
84B5:C8            REP     INY
84B6:10 F3                 BPL  PMES       ALWAYS
84B8:              *
84B8:              * TEXT
84B8:              *
84B8:84            TXT     DFB  $84
84B9:C2 D3 C1              ASC  /BSAV/
84BC:D6
84BD:45 84                 DFB  $45,$84
84BF:C2 CC CF              ASC  /BLOA/
84C2:C1
84C3:44                    DFB  $44
84C4:AC C1                 ASC  /,A/
84C6:24                    DFB  $24
84C7:AC CC                 ASC  /,L/
84C9:24                    DFB  $24
84CA:              *
84CA:              * END OF PROGRAM
84CA:              *

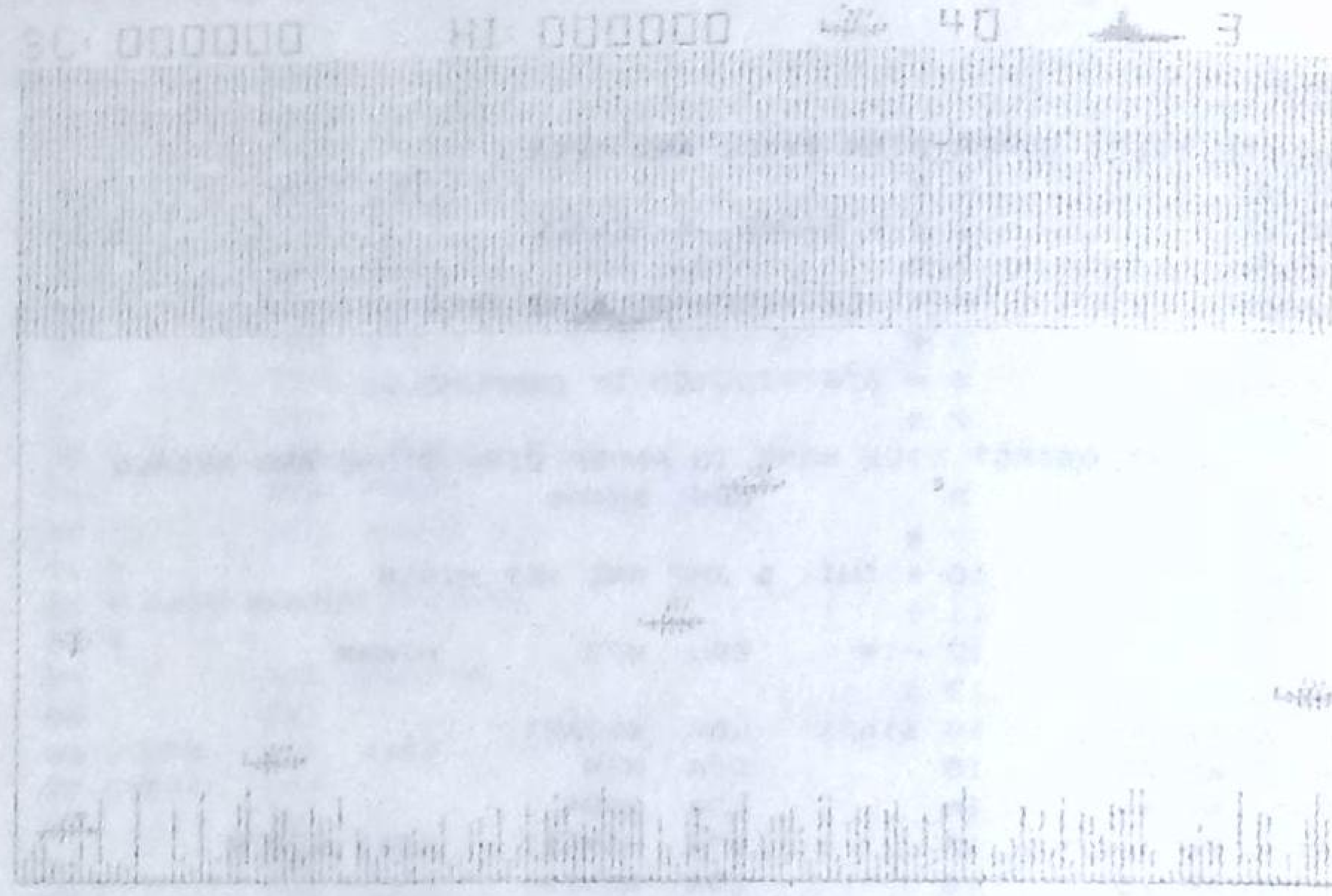
*** SUCCESSFUL ASSEMBLY: NO ERRORS
```

```
 03F5 BJP          F7C9 CALL         848B CEK            B1 CHNW
 8483 CHRN         DEBE CKOM         D66C CLEAR        DB5C COUT
 DAFB CROUT        E600 DR2          DB40 DR3          0A60 DS
 DD7B EEXP         8413 ENTS         F7D9 GNAME          73 HIM
 849D IEQ            FF MSK          842D NSTR         FDDA OHEX
 84A5 OM           D410 OMEM         845B PBLD         84AB PMES
 FF2D PRERR          84 PSTA         84B5 REP          848A RT1
   76 RUNB           9B SARA         8400 START          11 STTS
 DEC9 SYN          842A TPER         8497 TSM          84B8 TXT
 DD76 TYPM         D43C WBSIC        841E YST
```

```
   11 STTS           73 HIM            76 RUNB           84 PSTA
   9B SARA           B1 CHNW           FF MSK          03F5 BJP
 0A60 DS           8400 START        8413 ENTS         841E YST
 842A TPER         842D NSTR         845B PBLD         8483 CHRN
 848A RT1          848B CEK          8497 TSM          849D IEQ
 84A5 OM           84AB PMES         84B5 REP          84B8 TXT
 D410 OMEM         D43C WBSIC        D66C CLEAR        DAFB CROUT
 DB40 DR3          DB5C COUT         DD76 TYPM         DD7B EEXP
 DEBE CKOM         DEC9 SYN          E600 DR2          F7C9 CALL
 F7D9 GNAME        FDDA OHEX         FF2D PRERR
```

以GRAPHICS MAGICIAN 編寫的遊戲：DEPTH CHARGE

麥法基



筆者上一期已經詳細討論關於DEPTH CHARGE 的動畫程式部份，相信大家都已經把它造好而存放到磁碟上。這一期刊出遊戲的機械語言和 BASIC 部份。

無可否認遊戲效果越精細，程式必然越長。尤其是組合語語言部份，電腦是直接運行它所立即認識的語言，所以每一個指令，變數，旗號都要很清楚地加以編寫。電腦運行機械碼是根據指令立即運行，所以偵錯比較困難；因爲就算鍵入一個錯誤的指令或變數，程式仍然會若無其事的照樣運行，只不過效果無法預料而已。因此組合器是編寫遊戲程式的必須品，它除了有編輯功能外，更可以隨意更改任何變數及位址。

圖一是遊戲的主程式流程圖，由第 136 行開始至 211 行是主程式所在。主程式是控制遊戲運行的骨幹，它會分別跳往不同的副程式然後返回。首先讓我們看看各種不同副程式的功能。

㈠搖桿副程式（READPDL）

搖桿控制由第 347 行至 370 行的副程式負責。要讀取搖桿值並不困難，把 0 的值存放於 X— REGISTER 內用以選擇 PDL（0）作爲遊戲的搖桿控制。PREAD 是存放在ROM 內的一條副程式（$ FBIE），跳往 PREAD 後Y—REG—ISTER的值便是搖值了。

原則上我們可以直接把搖桿值直接存放到所需控制體的X及Y座標便可改變物體的位置，不過這樣不符合控制條件。遊戲的需求是我們可以控制戰艦以穩定速飛向左或右移動，很明顯地直接存放 PDL 值於X座標中戰艦移動速度會十分不規則。再者 PDL 值由 0 至 255 而戰艦移動是由預移圖序所產生，所以戰艦座標值跟本不能和 PDL 值互相配合。這種直接存放的方法我們並不採用。

比較簡單的方法去達成所要求效果是把 PDL 值間接影響物體的移動速度及方向。既然 PDL 的值由 0 至 255，我們可以把該值分爲三部份如下：

PDL 值	0—85	85—170	大於170
物體移動	向左	停止	向右

READPDL 這個程式便是把上列的值控制戰艦向不同方向移動的程式。程式解說如列表A所示。

請留意列表A第 357，363 及 369行$ 8994 是戰艦的路綫表所在位址。讀者請參考上一期關於路綫的問題，路綫表位址的值是影響物體移動方向及速度初第 356 行的$ 0 C是向左以三位元速度移動，第 362行的$ F F是靜止不動而第 368 行的$ 0 3 是向右以三位元速度移動。不過不要心急把所有值存入$ 8994內，首先利用上期的PATH ADDRESS FI—MDER 找出戰艦的位址，假如不是$8994 的話，則需把位址作適當修改。（你可能因輸入動畫程式時次序有所不同，就算次序不相同但仍然可以在主程式內修改。）

至於第 353 至 365 行是把戰艦局限於螢幕上移動，否則戰艦會移動出螢幕以外

```
347 READPDL  LDX #$00         ;SELECT PDL(0)
348          JSR PREAD        ;READ PDL VALUE
349          STY PDL          ;STORE INTO VARIABLE PDL
350          LDA PDL          ;LOAD BACK VALUE
351          CMP #$55         ;> DECIMAL 85?
352          BGE PDL.1        ;YES, TO PDL.1
353 PDL1     LDA $94A0        ;NO, TAKE VALUE OF SHIP'S X-BYTE POSITION
354          CMP #$0          ;EQUAL TO 0?
355          BEQ PDL.2        ;YES, BRANCH TO PDL.2
356          LDA #$0C         ;NO, LOAD VALUE #$0C
357          STA $8994        ;INTO SHIP'S PATH
358          RTS              ;RETURN
359 PDL.1    LDA PDL          ; LOAD PDL VALUE
360          CMP #$AA         ;> DEC 170?
361          BGE PDL2         ;YES, TO PDL2
362 PDL.2    LDA #$FF         ;NO, LOAD VALUE #$FF
363          STA $8994        ;INTO SHIP'S PATH
364          RTS              ;RETURN
365 PDL2     LDA $94A0        ;SHIP'S BYTE POSITION
366          CMP #$24         ;EQUAL TO DECIMAL 36?
367          BEQ PDL.2        ;YES, TO PDL.2
368          LDA #$03         ;NO, LOAD VALUE #$03
369          STA $8994        ;INTO SHIP'S PATH
370          RTS              ;RETURN
```

表 A

```
1305 XBOMB1    LDA $94CC      ;VERTICAL POSITION OF XBOMB
1306           CMP #$40       ;LESS THEN 64?
1307           BLT XBOMB.1    ;YES, TO XBOMB.1
1308           RTS            ;NO, CONTINUE MOVING
1309 XBOMB.1   LDA $94CA      ;HORIZONTAL BYTE VALUE OF XBOMB
1310           STA XBYTE      ;STORE TO XBYTE VARIABLE
1311           LDA $94CB      ;HORIZONTAL BIT VALUE OF XBOMB
1312           STA XBIT       ;STORE TO XBIT VARIABLE
1313           LDA $94CC      ;VERTICAL VALUE OF XBOMB
1314           STA Y          ;STORE TO Y VARIABLE
1315           LDA #$5        ;VALUE OF 5
1316           STA SHNUM      ;STORE TO SHAPE NUMBER
1317           JSR $8FA2      ;ERASE XBOMB
1318           LDA #$BF       ;VERTICAL VALUE OF BOTTOM OF SEA
1319           STA $94CC      ;STORE TO VERTICAL POSITION
1320           LDA $94A0      ;HORIZONTAL POSITION OF XBOMB
1321           CLC
1322           ADC XDIST1     ;ADDED WITH DISTANCE BETWEEN SHIP AND XBOMB
1323           STA $94CA      ;STORE BACK TO HORIZONTAL POSITION
1324           RTS            ;RETURN TO MAIN PROGRAM
```

表 B

```
1375 XBCHK     LDX #$00       ;INDEX=0
1376 XBCHK1    LDA $94C7,X    ;HORIZONTAL BYTE OF XBOMB
1377           CMP #$29       ;REACHS RIGHT MARGIN?
1378           BEQ XBCHK3     ;YES, TO XBCHK3
1379           CMP #$FB       ;NO, REACHS LEFT MARGIN?
1380           BEQ XBCHK4     ;YES, TO XBCHK4
1381 XBCHK2    INX            ;NO, CHECK ANOTHER ONE
1382           INX
1383           INX
1384           CPX #$A
1385           BLT XBCHK1     ;FINISHED CHECKING?
1386           RTS            ;NO, CHECK AGAIN
1387 XBCHK3    LDA #$FC       ;YES, RETURN
1388           STA $94C7,X    ;SET XBOMB TO LEFT MARGIN
1389           JMP XBCHK2     ;
1390 XBCHK4    LDA #$28       ;ANOTHER ONE
1391           STA $94C7,X    ;SET XBOMB TO RIGHT MARGIN
1392           JMP XBCHK2     ;
                              ;ANOTHER ONE
```

表 C

㈡移動水雷（XBOMBI 至XBOMB3）

畫面上總共有三枚水雷不斷從海底向水面移動。水雷移動的方向及速度由動畫程式所負責，而XBOMBI 等副程式則負責它們的位置，其解說如列表B所示。

DEPTH CHARGE

(三) 水雷水平位置程式（XBCHK）

這是把水雷水平位置固定在畫面上移動使它不會超過螢幕外的副程式：見列表C所示。

(四) 深水炸爆發射（CHARGE）

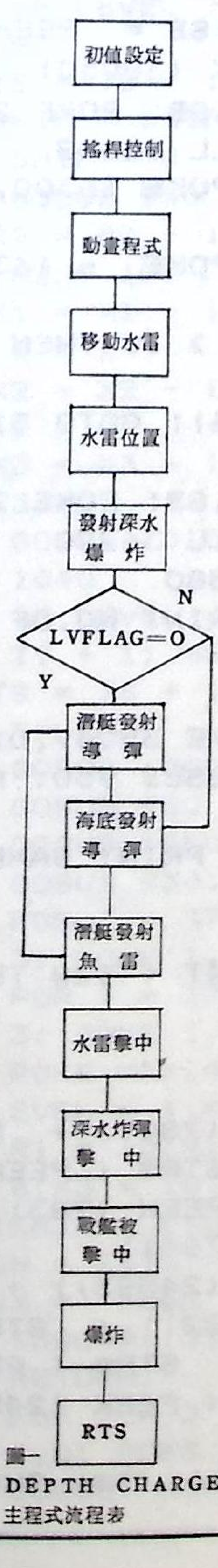
上列程式是水雷第一號之註解，其他兩枚同一原理。其中$94CA至$94CC是水雷之水平及平垂直置資料存放位址，第1315行的#$5是圖形號碼，而第1317行的$8FA2是消圖副程式，已經包括在動畫程式內。消圖原因是當水雷移動到水面時自動消失而再次由水底向上移動，造成一種不斷有水雷升上水面的效果。

列表D所所附的程式要留意的是聲音部份，由於APPLE聲音須要由程式控制，（其他電腦如ATARI的聲音和畫面是分劃開獨立進行）所以聲音發聲由CCOUNTI處理發聲次數。如此畫面移動才不會被發聲造成延運而產生不圓滑的效果。

以上幾個副程式代表了遊戲的幾個主要運行原理，明白程式的作用，其他部份自然容易明白了。

下期介紹另外一個更快速緊張的遊戲—— ZYTRO WAR 並且再討論其他問題如碰撞及不同路綫的運用，再見！

DEPTH CHARGE 主程式
SOURCE CODE LISTING
BACKGROUND MAKER □



圖一
DEPTH CHARGE
主程式流程表

```
CHARGE   LDA CHFLAG     ;CHARGE EXISTS?
         BNE CHAR1      ;YES, TO CHAR1 AND CONTINUE MOVING
         LDA BUTTON     ;NEGATIVE IF BUTTON PRESSED
         BMI FIRE1      ;BUTTON PRESSED, TO FIRE1
         LDA #$00       ;BUTTON NOT PRESSED
         STA LFLAG1     ;THEREFORE, SET FLAG TO 0
         JMP CHAR.1E    ;AND NO CHARGE FIRED
FIRE1    LDA LFLAG1     ;IS BUTTON BEING HELD DOWN?
         CMP #$01       ;
         BGE CHAR.1E    ;YES, NO CHARGE FIRED
         LDA CCFLAG1    ;BUTTON HAS BEEN RELEASED
         BNE CHAR.1E    ;IF LAST CHARGE EXPLODING, NOT FIRE YET
         LDA #$02       ;NO CHARGE EXPLODES, SO FIRE
         STA TIME       ;FIRE SOUND BEGINS
         LDA #$80       ;
         STA PITCH      ;
         JSR FSOUND     ;
         LDA #$01
         STA CCOUNT1    ;SET SOUND FLAG TO 1
         LDA $94A0      ;SET SHIP'S HORI BYTE VALUE TO
         STA $94D3      ;STARTING HORI. POSITION OF CHARGE
         LDA #$03       ;ALOS SET XBIT VALUE OF #$3
         STA $94D4      ;TO CHARGE'S XBIT
         LDA $94A2      ;FINALLY, SET SHIP'S VERT VALUE
         STA $94D5      ;TO STARTING VERT OF CHARGE
         LDA #$01       ;CHARGE NOW EXISTS
         STA CHFLAG     ;SET FLAG TO 1
         STA LFLAG1     ;INFORM APPLE THAT
         STA LFLAG2     ;BUTTON HAS BEEN HELD DOWN
         STA LFALG3     ;
         LDA #$00       ;STARING PATH NUMBER OF CHARGE
         STA $9473      ;IS NOW SET TO 0
         LDA #$88       ;ALSO SET PATH LOCATION
         STA $9474      ;TO THE BEGINNING OF MOVE
         LDA #$88
         STA $9475
         JMP CHAR.1     ;JMP TO CHAR.1 AND START MOVING
CHAR1    LDA $94D5      ;CHARGE'S VERT POSITION
         CMP #$C1       ;MOVE TO BOTTOM?
         BLT CHAR.1     ;NO, CONTINUE MOVING
         LDA #$0        ;YES, CHARGE NOW DOES NOT EXIST
         STA CHFLAG     ;SET FLAG TO 0
         LDA #$FE       ;STOP MOVING
         STA $9473      ;SET PATH NUMBER TO #$FE
         LDA #$D0       ;SET CHARGE'S VERT POSITION
         STA $94D5      ;TO THE TOP OF SEA
CHAR.1   LDA CCOUNT1    ;CHECK FIRE SOUND FLAG
         BEQ CHAR.1E    ;IF 0, STOP SOUND
         DEC CCOUNT1    ;IF NOT 0, DECREASE SOUND FLAG BY 1
         LDA #$05       ;CONTINUE PRODUCING SOUND
         STA TIME
         LDA #$12
         STA PITCH
         JSR FSOUND
CHAR.1E  RTS            ;RETURN TO MAIN PROGRAM
```

表 D

DEPTH CHARGE

BACKGROUND

```
0  REM : RUN THIS PROGRAM AND SAV
   E
1  REM : THE ENTIRRE BACKGROUND T
   O
2  REM : DISK WITH FILENAME "DEPT
   H
3  REM : CHARGE.PIC,A$4000,L$1FFF
   "
4  REM : ALSO ADD TEXT OF SCORE,
5  REM : HISCORE, ETC. AND SAVE
6  REM : WITH FILENAME "DEPTH
7  REM : CHARGE.PICI,A$4000,
8  REM : L$1FFF"
10  HGR : POKE  - 16302,0
15  HCOLOR= 5
20  FOR X = 0 TO 279
30 R =  INT ( RND (1) * 20 + 170)

40  HPLOT X,R TO X,190
50  NEXT
55  HCOLOR= 6
60  FOR Y = 10 TO 100
70  HPLOT 0,Y TO 279,7
80  NEXT
```

DEPTH CHARGE BASIC PROGRAM

```
5  HIMEM: 31608
6  TEXT : HOME
10  HGR2 : HGR : POKE  - 16302,0
20  PRINT  CHR$ (4);"BLOAD DEPTH
    CHARGE.PIC1"
30  PRINT  CHR$ (4);"BLOAD DEPTH
    CHARGE.PIC,A$4000"
40  PRINT  CHR$ (4);"BLOAD DEPTH
    CHARGE.OB,A$6000"
50  PRINT  CHR$ (4);"BLOAD DEPTH
    CHARGE.ANM"
59  REM  **** TITLE ****
60  POKE  - 16368,0
70  IF  PEEK ( - 16384) > 127 THEN
    90
80  GOTO 70
90  HCOLOR= 6
100  FOR Y = 20 TO 60: HPLOT 0,Y TO
     279,Y: NEXT
120  HCOLOR= 0
130  FOR Y = 90 TO 160: HPLOT 0,Y
      TO 279,Y: NEXT
140  REM  *** THEME ***
145  POKE 234,0: GOSUB 800: REM
     INITIALISATION
150  GOSUB 10000: REM  PRINT LEVE
     L
160  GOSUB 900: REM  PRINT SCORE
162  GOSUB 904: REM  INIT HIGH SC
     ORE
165  GOSUB 12000: REM  PRINT HIGH
      SCORE
170  GOSUB 930: REM  PRINT NOS. O
     F SUBMARINE
180  GOSUB 940: REM  PRINT NOS. O
     F SHIPS
190  GOSUB 950: REM  SET SUBMARIN
     E'S SPEED
200  GOSUB 1010: REM  SET TOPPEDO
     'S ACCURACY
210  GOSUB 1020: REM  SET ACCURAC
     Y OF RBOMB & XBOMB
220  GOSUB 1050: REM  SET SPEED O
     F TOPPEDO
230  GOSUB 1040: REM  SET POSITIO
     N OF SUBMARINE
240  POKE 255,0: REM  SET RTN COD
     E TO 0
250  CALL 24576: REM  BEGIN GAME
260  ON  PEEK (255) GOSUB 300,400
270  GOTO 240
300 SHIP = SHIP - 1:EX = 0
302 SX =  PEEK (38048):SB =  PEEK
     (38049):SY =  PEEK (38050)
305  POKE 0,SX: POKE 1,SB: POKE 2
     ,SY: POKE 3,7: CALL 36608
310  POKE  - 16303,0: POKE 16300,
     0
320  POKE  - 16304,0: POKE  - 163
     02,0
330 EX = EX + 1: IF EX > 10 THEN
     350
340 S =  PEEK ( - 16336): GOTO 31
     0
350  POKE 0,SX: POKE 1,SB: POKE 2
     ,SY: POKE 3,7: CALL 36770
360  IF SHIP < 0 THEN 380
365  GOSUB 940: REM  PRINT NO.OF
     SHIPS
368  GOSUB 1050
370  POKE 38048,20: POKE 38049,0:
      POKE 38050,63: GOSUB 950: RETURN
380  GOSUB 10200: REM  PRINT GAME
     OVER
382  POKE  - 16368,0
384  IF  PEEK ( - 16384) < 128 THEN
     384
386  GOSUB 13000
388  POKE 234,0
389 B$ =  STR$ ( PEEK (780)) +  STR$
     ( PEEK (781)) +  STR$ ( PEEK
     (782)) +  STR$ ( PEEK (783))
      +  STR$ ( PEEK (784))
390 A$ =  STR$ ( PEEK (24583)) +
      STR$ ( PEEK (24582)) +  STR$
     ( PEEK (24581)) +  STR$ ( PEEK
     (24580)) +  STR$ ( PEEK (245
     79))
391  IF  VAL (A$) >  VAL (B$) THEN
```

DEPTH CHARGE

```
     POKE 780, PEEK (24583): POKE
    781, PEEK (24582): POKE 782,
     PEEK (24581): POKE 783, PEEK
    (24580): POKE 784, PEEK (245
    79): POKE 785,0
392 GOSUB 800: GOSUB 10000: GOSUB
    900: GOSUB 12000: GOSUB 930:
     GOSUB 940: GOSUB 950: GOSUB
    1010: GOSUB 1020: GOSUB 1050
    : GOSUB 1040
395 RETURN
400 LEVEL = LEVEL + 1:SN = SN + 1
    : IF SN > 4 THEN SN = 4
401 LL = LL + 1: IF LL = 10 THEN
    LL = 0
402  IF LEVEL > 5 THEN  POKE 234,
    1
404  IF LEVEL > 9 THEN TENTH = 1
405  IF LEVEL > 19 THEN TENTH = 2
406  IF LEVEL > 29 THEN LEVEL = 2
    9
410 TS = TS + 1
415 SHIP = SHIP + 1: IF SHIP > 5 THEN
    SHIP = 5
416  GOSUB 940
420 R2 = R2 - 1: IF R2 < 10 THEN
    R2 = 10
422 X1 = X1 - 1: IF X1 < 2 THEN X
    1 = 2
424 X2 = X2 - 1: IF X2 < 4 THEN X
    2 = 4
426 X3 = X3 - 1: IF X3 < 2 THEN X
    3 = 2
430  GOSUB 1010: GOSUB 1020: GOSUB
    1040
440  IF LEVEL < 3 THEN TS = 0:TT =
    TT + 1: GOTO 470
450 TS = TS + 1: IF TS > 11 THEN
    TS = 11
460  GOSUB 1300: GOTO 472
470  GOSUB 950
472  GOSUB 1050
475  GOSUB 930: GOSUB 10000
480  FOR I = 37955 TO 37970 STEP
    3: POKE I,0: NEXT
490  FOR I = 37994 TO 38000 STEP
    3: POKE I,0: NEXT
500  POKE MAX,40: RETURN
800 LEVEL = 1:MAX = 210:SHIP = 3:
    R1 = 10:R2 = 16:X1 = 12:X2 =
    6:X3 = 10
810 TENTH = 0:LL = 1
820 SN = 2:TD = 12:TS = 0:TT = 0
830 SX =  PEEK (38048):SB =  PEEK
    (38049):SY =  PEEK (38050)
840  RETURN
900  FOR I = 3 TO 8: POKE 0,I: POKE
    1,0: POKE 2,0: POKE 3,9: CALL
    36608: NEXT
```

```
902  FOR I = 24579 TO 24583: POKE
    I,0: NEXT
903  RETURN
904  FOR I = 780 TO 785: POKE I,0
    : NEXT
920  RETURN
930  POKE MAX,40: POKE 24584,0: POKE
    24585,4: POKE 0,28: POKE 1,0
    : POKE 2,0: POKE 3,13: CALL
    36608: POKE 0,29: POKE 1,0: POKE
    2,0: POKE 3,9: CALL 36608: RETURN
940  POKE 0,36: POKE 1,0: POKE 2,
    0: POKE 3,SHIP + 9: CALL 366
    08: RETURN
950  FOR I = 35208 TO 35218 STEP
    2:SS =  INT ( RND (1) * SN)
960  IF SS = 0 THEN  POKE I,2: GOTO
    990
970  IF SS = 1 THEN  POKE I,8: GOTO
    990
975  IF SS = 2 THEN  POKE I,3: GOTO
    990
980  POKE I,12
990  NEXT
1000  RETURN
1010  FOR I = 768 TO 772:RT =  INT
    ( RND (1) * TD): POKE I,RT: NEXT
1015  RETURN
1020  POKE 773,R1: POKE 774,R2: POKE
    775,X1: POKE 776,X2: POKE 77
    7,X3: RETURN
1040  FOR I = 38051 TO 38066 STEP
    3:RS =  INT ( RND (1) * 40):
    POKE I,RS: NEXT : RETURN
1050  FOR I = 35002 TO 35010 STEP
    2
1060 RR =  INT ( RND (1) * TS) +
    TT
1070  IF RR = 0 THEN  POKE I,64: GOTO
    1090
1071  IF RR = 1 THEN  POKE I,128:
    GOTO 1090
1072  IF RR = 2 THEN  POKE I,192:
    GOTO 1090
1073  IF RR = 3 THEN  POKE I,132:
    GOTO 1090
1074  IF RR = 4 THEN  POKE I,129:
    GOTO 1090
1075  IF RR = 5 THEN  POKE I,196:
    GOTO 1090
1076  IF RR = 6 THEN  POKE I,193:
    GOTO 1090
1077  IF RR = 7 THEN  POKE I,200:
    GOTO 1090
1078  IF RR = 8 THEN  POKE I,194:
    GOTO 1090
1079  IF RR = 9 THEN  POKE I,195:
    GOTO 1090
```

DEPTH CHARGE

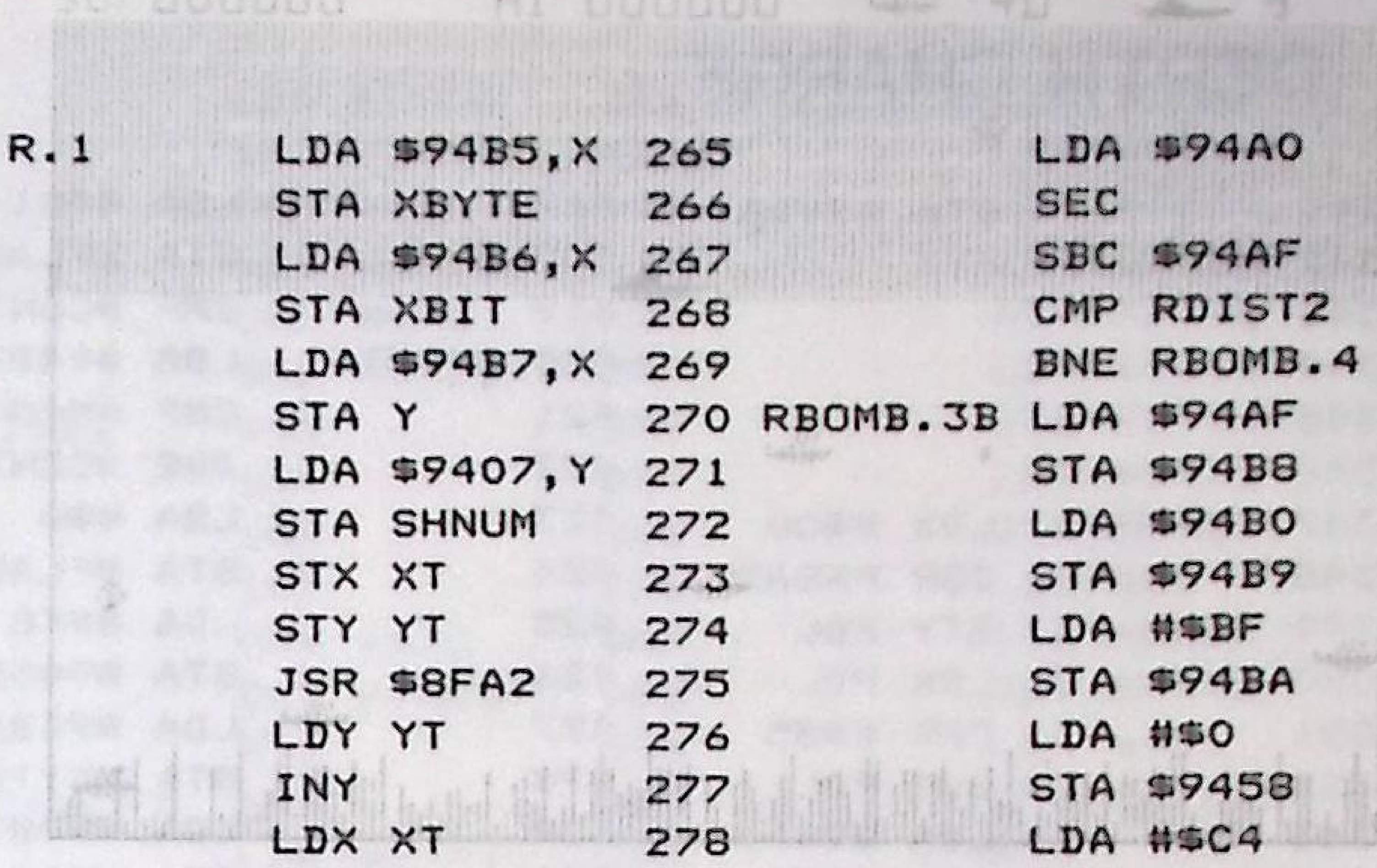
```
1080  POKE I,204
1090  POKE I + 1,0
1095  NEXT I
1100  RETURN
1200  FOR I = 35002 TO 35010 STEP
   2:RR =  INT ( RND (1) * 3)
1210  IF RR = 0 THEN  POKE I,64: GOTO
   1240
1220  IF RR = 1 THEN  POKE I,128:
   GOTO 1240
1230  POKE I,192
1240  POKE I + 1,0
1245  NEXT I
1250  RETURN
1300  FOR I = 35208 TO 35218 STEP
   2:SS =  INT ( RND (1) * 2)
1310  IF SS = 0 THEN  POKE I,3: GOTO
   1330
1320  POKE I,12
1330  POKE I + 1,0
1340  NEXT I
1350  RETURN
9999  REM  PRINT LEVEL ROUTINE
10000 X = 90:Y = 80:Z = 26: GOSUB
   10100
10005 X = 98:Y = 80:Z = 27: GOSUB
   10100
10010 X = 108:Y = 80:Z = 19: GOSUB
   10100
10015 X = 119:Y = 80:Z = 22: GOSUB
   10100
10020 X = 129:Y = 80:Z = 21: GOSUB
   10100
10024 X = 151:Y = 80:Z = TENTH +
   9: GOSUB 10100
10025 X = 162:Y = 80:Z = LL + 9: GOSUB
   10100
10030 X = 98:Y = 100:Z = 25: GOSUB
   10100
10035 X = 108:Y = 100:Z = 21: GOSUB
   10100
10040 X = 119:Y = 100:Z = 19: GOSUB
   10100
10045 X = 130:Y = 100:Z = 20: GOSUB
   10100
10050 X = 140:Y = 100:Z = 29: GOSUB
   10100
10055 X = 150:Y = 100:Z = 30: GOSUB
   10100
10060  FOR I = 1 TO 400: NEXT
10070  HCOLOR= 0
10080  FOR X = 80 TO 200: HPLOT X
   ,80 TO X,110: NEXT
10090  RETURN
10100 S =  PEEK ( - 16336) +  PEEK
   ( - 16336) +  PEEK ( - 16336
   )
10120  POKE 0, INT (X / 7): POKE
   1,X -  INT (X / 7) * 7: POKE
   2,Y: POKE 3,Z: CALL 36608: RETURN

10199  REM  ** GAME OVER **
10200 X = 40:Y = 80:Z = 22: GOSUB
   10100
10205 X = 60:Y = 80:Z = 19: GOSUB
   10100
10300 X = 80:Y = 80:Z = 23: GOSUB
   10100
10305 X = 100:Y = 80:Z = 21: GOSUB
   10100
10400 X = 160:Y = 80:Z = 24: GOSUB
   10100
10405 X = 180:Y = 80:Z = 28: GOSUB
   10100
10500 X = 200:Y = 80:Z = 21: GOSUB
   10100
10505 X = 220:Y = 80:Z = 25: GOSUB
   10100
10600  RETURN
12000 N = 16
12010  FOR I = 780 TO 785
12020  POKE 0,N: POKE 1,0: POKE 2
   ,0: POKE 3, PEEK (I) + 9: CALL
   36608
12030 N = N + 1
12040  NEXT I
12050  RETURN
13000  HCOLOR= 0
13010  FOR I = 40 TO 240: HPLOT I
   ,80 TO I,100: NEXT
13020  FOR I = 38051 TO 38066 STEP
   3
13030  POKE 0, PEEK (I): POKE 1, PEEK
   (I + 1): POKE 2, PEEK (I + 2
   ): POKE 3,1: CALL 36770
13040  NEXT I
13050  RETURN □
```

DEPTH CHARGE SOURCE CODE

```
         ORG $6000
         OBJ $800
         JMP BEGIN
;
;
XBYTE    EPZ $0
XBIT     EPZ $1
Y        EPZ $2
SHNUM    EPZ $3
TIME     EPZ $D0
PITCH    EPZ $D1
MAX      EPZ $D2
XLEFT    EPZ $10
XRIGHT   EPZ $11
PDL      EPZ $12
LVLFLG   EPZ $EA
RTN      EPZ $FF
;
PREAD    EQU $FB1E
BUTTON   EQU $C061
BDIST1   EQU $300
BDIST2   EQU $301
BDIST3   EQU $302
BDIST4   EQU $303
BDIST5   EQU $304
RDIST1   EQU $305
RDIST2   EQU $306
XDIST1   EQU $307
XDIST2   EQU $308
XDIST3   EQU $309
SCORE0   DFS 1
SCORE1   DFS 1
SCORE2   DFS 1
SCORE3   DFS 1
SCORE4   DFS 1
MAXSC1   DFS 1
```

DEPTH CHARGE



```
MAXSC2    DFS 1
XRIGHT1   DFS 1
XLEFT1    DFS 1
XT        DFS 1
YT        DFS 1
XBFLAG1   DFS 1
XBFLAG2   DFS 1
XBFLAG3   DFS 1
FLAG      DFS 1
FLAG1     DFS 1
BFLAG1    DFS 1
BFLAG2    DFS 1
BFLAG3    DFS 1
BFLAG4    DFS 1
BFLAG5    DFS 1
CHFLAG    DFS 1
CHFLAG2   DFS 1
CHFLAG3   DFS 1
CLFLAG    DFS 1
LFLAG1    DFS 1
LFLAG2    DFS 1
LFLAG3    DFS 1
CCOUNT1   DFS 1
CCOUNT2   DFS 1
CCOUNT3   DFS 1
CHCOUNT1  DFS 1
CHCOUNT2  DFS 1
CHCOUNT3  DFS 1
CCFLAG1   DFS 1
CCFLAG2   DFS 1
CCFLAG3   DFS 1
CBYTE1    DFS 1
CBYTE2    DFS 1
CBYTE3    DFS 1
CBIT1     DFS 1
CBIT2     DFS 1
CBIT3     DFS 1
CY1       DFS 1
CY2       DFS 1
CY3       DFS 1
AYTEMP    DFS 1
BYTEMP    DFS 1
CYTEMP    DFS 1
NUMBER    DFS 1
;
;
BEGIN     LDA #$0
          STA FLAG
          STA FLAG1
          STA BFLAG1
          STA BFLAG3
          STA BFLAG4
          STA CHFLAG
          STA BFLAG2
          STA CHFLAG2
          STA CHFLAG3
          STA CLFLAG
          STA CCFLAG1
          STA CCFLAG2
          STA CCFLAG3
          LDA #$6
          STA NUMBER
          LDA #$FE
          STA $9455
          STA $9458
          STA $945B
          STA $945E
          STA $9461
          STA $9464
          STA $9467
          STA $9473
          STA $9476
          STA $9479
          LDA #$D0
          STA $94D5
          STA $94D8
          STA $94DB
          LDA #$BF
          STA $94B7
          STA $94BA
          STA $94CC
          STA $94CF
          STA $94D2
          LDA #$7C
          STA $9456
          LDA #$87
          STA $9457
          LDA #$19
          STA $9459
          LDA #$87
          STA $945A
          LDA #$29
          STA XRIGHT
          LDA #$28
          STA XRIGHT1
          LDA #$F8
          STA XLEFT
          LDA #$F9
          STA XLEFT1
START     JSR READPDL
          JSR $9040
          JSR XBOMB1
          JSR XBOMB2
          JSR XBOMB3
          JSR XBCHK
          JSR CHARGE
          JSR CHARGE2
          JSR CHARGE3
          LDA LVLFLG
          BNE START.1
          JSR RBOMB1
          JSR RBOMB2
          JMP START.1A
START.1   JSR RBOMBH1
          JSR RBOMBH2
START.1A  JSR BOMB1
          JSR BOMB2
          JSR BOMB3
          JSR BOMB4
          JSR BOMB5
          JSR XBCOL
          JSR CHCL1
          JSR CHCL2
          JSR CHCL3
          JSR CLCHECK
          LDA CLFLAG
          BNE RETURN
          LDA CCFLAG1
          BEQ START.2
          JSR EXPLO1
START.2   LDA CCFLAG2
          BEQ START.3
          JSR EXPLO2
START.3   LDA CCFLAG3
          BEQ START.4
          JSR EXPLO3
          JMP LOOPA
START.4   LDA MAX
          BEQ RETURN1
LOOPA     LDA MAX
          CMP #$6
          BGE LOOP
          JSR DELAY
LOOP      JSR CHKBOUND
          JMP START
RETURN    LDA #$01
          STA RTN
          JMP R
RETURN1   LDA #$2
          STA RTN
R         LDX #$00
          LDY #$00
R.1       LDA $94B5,X
          STA XBYTE
          LDA $94B6,X
          STA XBIT
          LDA $94B7,X
          STA Y
          LDA $9407,Y
          STA SHNUM
          STX XT
          STY YT
          JSR $8FA2
          LDY YT
          INY
          LDX XT
          INX
          INX
          INX
          CPX #$25
          BLT R.1
          JSR EX1.2
          JSR EX2.2
          JSR EX3.2
          RTS
;
;
;
RBOMB1    LDA $9455
          CMP #$FE
          BEQ RBOMB.1
          JMP RBOMB.2A
RBOMB.1   LDA $94B5
          STA XBYTE
          LDA $94B6
          STA XBIT
          LDA $94B7
          STA Y
          LDA #$4
          STA SHNUM
          JSR $8FA2
RBOMB.1A  LDA $94AC
          SEC
          SBC $94A0
          CMP RDIST1
          BNE RBOMB.2
RBOMB.1B  LDA $94AC
          STA $94B5
          LDA $94AD
          STA $94B6
          LDA #$BF
          STA $94B7
          LDA #$0
          STA $9455
          LDA #$27
          STA $9456
          LDA #$89
          STA $9457
          JMP RBOMB.2A
RBOMB.2   LDA #$64
          STA $94B5
RBOMB.2A  RTS
;
;
;
RBOMB2    LDA $9458
          CMP #$FE
          BEQ RBOMB.3
          JMP RBOMB.4A
RBOMB.3   LDA $94B8
          STA XBYTE
          LDA $94B9
          STA XBIT
          LDA $94BA
          STA Y
          LDA #$4
          STA SHNUM
          JSR $8FA2
          LDA $94A0
          SEC
          SBC $94AF
          CMP RDIST2
          BNE RBOMB.4
RBOMB.3B  LDA $94AF
          STA $94B8
          LDA $94B0
          STA $94B9
          LDA #$BF
          STA $94BA
          LDA #$0
          STA $9458
          LDA #$C4
          STA $9459
          LDA #$88
          STA $945A
          JMP RBOMB.4A
RBOMB.4   LDA #$78
          STA $94B8
RBOMB.4A  RTS
;
;
RBOMBH1   LDA $9455
          CMP #$FE
          BEQ RBOMBH.1
          JMP RBOMBH.2
RBOMBH.1  LDA $94B5
          STA XBYTE
          LDA $94B6
          STA XBIT
          LDA $94B7
          STA Y
          LDA #$4
          STA SHNUM
          JSR $8FA2
          LDA #$0
          STA $9455
          LDA #$27
          STA $9456
          LDA #$89
          STA $9457
          LDA $94A0
          CLC
          ADC RDIST1
          STA $94B5
          LDA #$BF
          STA $94B7
RBOMBH.2  RTS
;
;
RBOMBH2   LDA $9458
          CMP #$FE
          BEQ RBOMBH.3
          JMP RBOMBH.4
RBOMBH.3  LDA $94B8
          STA XBYTE
          LDA $94B9
          STA XBIT
          LDA $94BA
          STA Y
          LDA #$4
          STA SHNUM
          JSR $8FA2
          LDA #$0
          STA $9458
          LDA #$C4
          STA $9459
          LDA #$88
          STA $945A
          LDA $94A0
          SEC
          SBC RDIST2
          STA $94B8
          LDA #$BF
```

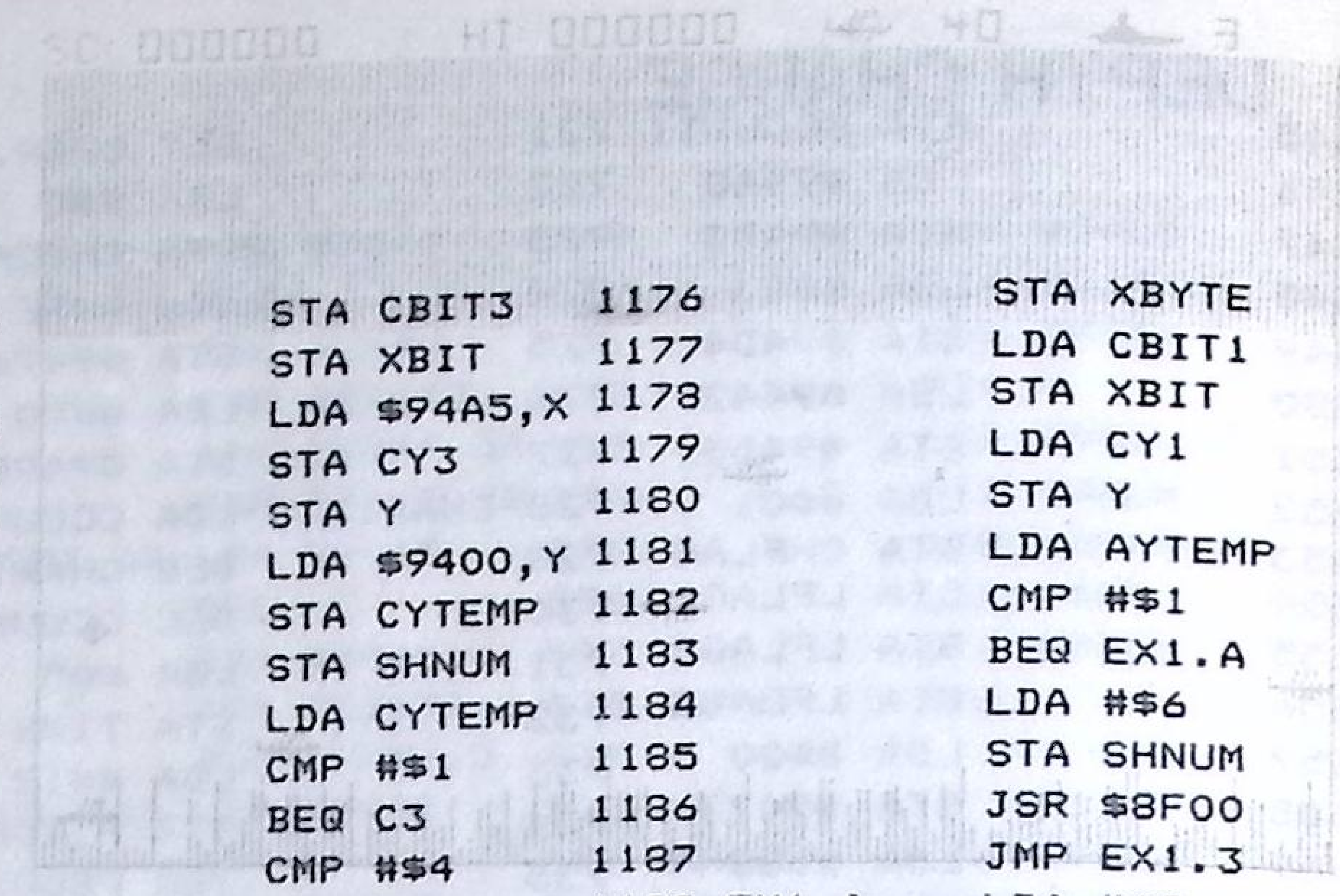
DEPTH CHARGE

```
         STA $94BA
RBOMBH.4 RTS
;
;
;
;
READPDL  LDX #$00
         JSR PREAD
         STY PDL
         LDA PDL
         CMP #$55
         BGE PDL.1
PDL1     LDA $94A0
         CMP #$0
         BEQ PDL.2
         LDA #$0C
         STA $8994
         RTS
PDL.1    LDA PDL
         CMP #$AA
         BGE PDL2
PDL.2    LDA #$FF
         STA $8994
         RTS
PDL2     LDA $94A0
         CMP #$24
         BEQ PDL.2
         LDA #$03
         STA $8994
         RTS
;
;
;
CHKBOUND LDX #$00
CHKBD1   LDA $94A3,X
         CMP XRIGHT
         BEQ SKIP1
         CMP XLEFT
         BEQ SKIP2
CHKBD2   INX
         INX
         INX
         CPX #$11
         BLT CHKBD1
         RTS
SKIP1    LDA XLEFT1
         STA $94A3,X
         JMP CHKBD2
SKIP2    LDA XRIGHT1
         STA $94A3,X
         JMP CHKBD2
;
;
;
BOMB1    LDA $94BB
         CMP #$C8
         BEQ BOMB1A
         LDA BFLAG1
         BNE BCONT1
BOMB1A   LDA $94A3
         SEC
         SBC $94A0
         CMP BDIST1
         BNE BCONT.1A
         JMP BCONT.1B
BCONT.1A LDA $94A0
         SEC
         SBC $94A3
         CMP BDIST1
         BNE BCONT.1
BCONT.1B LDA $94A3
         STA $94BB
         LDA $94A5
         STA $94BD
         LDA #$0
         STA $945B
         LDA #$01
         STA BFLAG1
         JMP BCONT.1
BCONT1   LDA $94BD
         CMP #$40
         BGE BCONT.1
         LDA #$0
         STA BFLAG1
         LDA #$FE
         STA $945B
         LDA $94BB
         STA XBYTE
         LDA $94BC
         STA XBIT
         LDA $94BD
         STA Y
         LDA #$02
         STA SHNUM
         JSR $8FA2
         LDA #$C1
         STA $94BD
BCONT.1  RTS
;
;
;
BOMB2    LDA $94BE
         CMP #$C8
         BEQ BOMB2A
         LDA BFLAG2
         BNE BCONT2
BOMB2A   LDA $94A0
         SEC
         SBC $94A6
         CMP BDIST2
         BNE BCONT.2A
         JMP BCONT.2B
BCONT.2A LDA $94A0
         SEC
         SBC $94A6
         CMP BDIST2
         BNE BCONT.2
BCONT.2B LDA $94A6
         STA $94BE
         LDA $94A8
         STA $94C0
         LDA #$0
         STA $945E
         LDA #$01
         STA BFLAG2
         JMP BCONT.2
BCONT2   LDA $94C0
         CMP #$40
         BGE BCONT.2
         LDA #$0
         STA BFLAG2
         LDA #$FE
         STA $945E
         LDA $94BE
         STA XBYTE
         LDA $94BF
         STA XBIT
         LDA $94C0
         STA Y
         LDA #$02
         STA SHNUM
         JSR $8FA2
         LDA #$C1
         STA $94C0
BCONT.2  RTS
;
;
BOMB3    LDA $94C1
         CMP #$C8
         BEQ BOMB3A
         LDA BFLAG3
         BNE BCONT3
BOMB3A   LDA $94AC
         SEC
         SBC $94A0
         CMP BDIST3
         BNE BCONT.3A
         JMP BCONT.3B
BCONT.3A LDA $94A0
         SEC
         SBC $94AC
         CMP BDIST3
         BNE BCONT.3
BCONT.3B LDA $94AC
         STA $94C1
         LDA $94AE
         STA $94C3
         LDA #$0
         STA $9461
         LDA #$01
         STA BFLAG3
         JMP BCONT.3
BCONT3   LDA $94C3
         CMP #$40
         BGE BCONT.3
         LDA #$0
         STA BFLAG3
         LDA #$FE
         STA $9461
         LDA $94C1
         STA XBYTE
         LDA $94C2
         STA XBIT
         LDA $94C3
         STA Y
         LDA #$02
         STA SHNUM
         JSR $8FA2
         LDA #$C1
         STA $94C3
BCONT.3  RTS
;
;
;
BOMB4    LDA $94C4
         CMP #$C8
         BEQ BOMB4A
         LDA BFLAG4
         BNE BCONT4
BOMB4A   LDA $94AF
         SEC
         SBC $94A0
         CMP BDIST4
         BNE BCONT.4A
         JMP BCONT.4B
BCONT.4A LDA $94A0
         SEC
         SBC $94AF
         CMP BDIST4
         BNE BCONT.4
BCONT.4B LDA $94AF
         STA $94C4
         LDA $94B1
         STA $94C6
         LDA #$0
         STA $9464
         LDA #$01
         STA BFLAG4
         JMP BCONT.4
BCONT4   LDA $94C6
         CMP #$40
         BGE BCONT.4
         LDA #$0
         STA BFLAG4
         LDA #$FE
         STA $9464
         LDA $94C4
         STA XBYTE
         LDA $94C5
         STA XBIT
         LDA $94C6
         STA Y
         LDA #$02
         STA SHNUM
         JSR $8FA2
         LDA #$C1
         STA $94C6
BCONT.4  RTS
;
;
BOMB5    LDA $94C7
         CMP #$C8
         BEQ BOMB5A
         LDA BFLAG5
         BNE BCONT5
BOMB5A   LDA $94A9
         SEC
         SBC $94A0
         CMP BDIST5
         BNE BCONT.5A
         JMP BCONT.5B
BCONT.5A LDA $94A0
         SEC
         SBC $94A9
         CMP BDIST5
         BNE BCONT.5
BCONT.5B LDA $94A9
         STA $94C7
         LDA $94AB
         STA $94C9
         LDA #$0
         STA $9467
         LDA #$01
         STA BFLAG5
         JMP BCONT.5
BCONT5   LDA $94C9
         CMP #$40
         BGE BCONT.5
         LDA #$0
         STA BFLAG5
         LDA #$FE
         STA $9467
         LDA $94C7
         STA XBYTE
         LDA $94C8
         STA XBIT
         LDA $94C9
         STA Y
         LDA #$02
         STA SHNUM
         JSR $8FA2
         LDA #$C1
         STA $94C9
BCONT.5  RTS
;
;
CHARGE   LDA CHFLAG
         BNE CHAR1
         LDA BUTTON
         BMI FIRE1
         LDA #$00
         STA LFLAG1
         JMP CHAR.1E
FIRE1    LDA LFLAG1
         CMP #$01
         BGE CHAR.1E
         LDA CCFLAG1
         BNE CHAR.1E
         LDA #$02
         STA TIME
         LDA #$80
         STA PITCH
         JSR FSOUND
         LDA #$01
```

DEPTH CHARGE

```
         STA CCOUNT1
         LDA $94A0
         STA $94D3
         LDA #$03
         STA $94D4
         LDA $94A2
         STA $94D5
         LDA #$01
         STA CHFLAG
         STA LFLAG1
         STA LFLAG2
         STA LFLAG3
         LDA #$00
         STA $9473
         LDA #$88
         STA $9474
         LDA #$88
         STA $9475
         JMP CHAR.1
CHAR1    LDA $94D5
         CMP #$C1
         BLT CHAR.1
         LDA #$0
         STA CHFLAG
         LDA #$FE
         STA $9473
         LDA #$D0
         STA $94D5
CHAR.1   LDA CCOUNT1
         BEQ CHAR.1E
         DEC CCOUNT1
         LDA #$05
         STA TIME
         LDA #$12
         STA PITCH
         JSR FSOUND
CHAR.1E  RTS
;
CHARGE2  LDA CHFLAG2
         BNE CHAR2
         LDA BUTTON
         BMI FIRE2
         LDA #$00
         STA LFLAG2
         JMP CHAR.2E
FIRE2    LDA LFLAG2
         CMP #$01
         BGE CHAR.2E
         LDA CCFLAG2
         BNE CHAR.2E
         LDA #$2
         STA TIME
         LDA #$80
         STA PITCH
         JSR FSOUND
         LDA #$01
         STA CCOUNT2
         LDA $94A0
         STA $94D6
         LDA #$05
         STA $94D7
         LDA $94A2
         STA $94D8
         LDA #$00
         STA $9476
         LDA #$7B
         STA $9477
         LDA #$88
         STA $9478
         LDA #$01
         STA CHFLAG2
         STA LFLAG2
         STA LFLAG3
         JMP CHAR.2
CHAR2    LDA $94D8
         CMP #$C1
         BLT CHAR.2
         LDA #$0
         STA CHFLAG2
         LDA #$FE
         STA $9476
         LDA #$D0
         STA $94D8
CHAR.2   LDA CCOUNT2
         BEQ CHAR.2E
         DEC CCOUNT2
         LDA #$5
         STA TIME
         LDA #$12
         STA PITCH
         JSR FSOUND
CHAR.2E  RTS
;
;
CHARGE3  LDA CHFLAG3
         BNE CHAR3
         LDA BUTTON
         BMI FIRE3
         LDA #$00
         STA LFLAG3
         JMP CHAR.3E
FIRE3    LDA LFLAG3
         CMP #$01
         BGE CHAR.3E
         LDA CCFLAG3
         BNE CHAR.3E
         LDA #$2
         STA TIME
         LDA #$80
         STA PITCH
         JSR FSOUND
         LDA #$01
         STA CCOUNT3
         LDA $94A0
         STA $94D9
         LDA #$06
         STA $94DA
         LDA $94A2
         STA $94DB
         LDA #$00
         STA $9479
         LDA #$6E
         STA $947A
         LDA #$88
         STA $947B
         LDA #$01
         STA CHFLAG3
         STA LFLAG3
         JMP CHAR.3
CHAR3    LDA $94DB
         CMP #$C1
         BLT CHAR.3
         LDA #$0
         STA CHFLAG3
         LDA #$FE
         STA $9479
         LDA #$D0
         STA $94DB
CHAR.3   LDA CCOUNT3
         BEQ CHAR.3E
         DEC CCOUNT3
         LDA #$05
         STA TIME
         LDA #$12
         STA PITCH
         JSR FSOUND
CHAR.3E  RTS
;
;
CLCHECK  LDX #$00
         LDY #$00
FIND1    SEC
         LDA $94A0
         SBC $94B5,X
         CMP #$0
         BEQ ACHECK1
         CMP #$FE
         BGE ACHECK1
FIND2    INX
         INX
         INX
         INY
         CPX #$1C
         BLT FIND1
         RTS
ACHECK1  SEC
         LDA $94A2
         SBC $94B7,X
         CMP #$1
         BGE ACHECK2
         JMP ACHECK3
ACHECK2  CMP #$FE
         BLT FIND2
ACHECK3  LDA $94B5,X
         STA XBYTE
         LDA $94B6,X
         STA XBIT
         LDA $94B7,X
         STA Y
         LDA $9407,Y
         STA SHNUM
         JSR $8FA2
         LDA #$C1
         STA $94B7,X
         LDA $94A0
         STA XBYTE
         LDA $94A1
         STA XBIT
         LDA $94A2
         STA Y
         LDA #$0
         STA SHNUM
         JSR $8FA2
         LDA #$01
         STA CLFLAG
         RTS
;
XBCOL    LDX #$00
XFIND1   SEC
         LDA $94A0
         SBC $94CA,X
         CMP #$0
         BEQ XCHECK1
         CMP #$FE
         BGE XCHECK1
XFIND2   INX
         INX
         INX
         CPX #$7
         BLT XFIND1
         RTS
XCHECK1  SEC
         LDA $94A2
         SBC $94CC,X
         CMP #$1
         BGE XCHECK2
         JMP XCHECK3
XCHECK2  CMP #$FE
         BLT XFIND2
XCHECK3  LDA $94CA,X
         STA XBYTE
         LDA $94CB,X
         STA XBIT
         LDA $94CC,X
         STA Y
         LDA #$05
         STA SHNUM
         JSR $8FA2
         LDA #$C1
         STA $94CC,X
         LDA $94A0
         STA XBYTE
         LDA $94A1
         STA XBIT
         LDA $94A2
         STA Y
         LDA #$0
         STA SHNUM
         JSR $8FA2
         LDA #$01
         STA CLFLAG
         RTS
;
CHCL1    LDA CCFLAG1
         BEQ CHCL1.1
         JMP CH1END
CHCL1.1  LDX #$00
         LDY #$1
FINDA1   STX XT
         LDA $94D3,X
         SBC $94A3,X
         CMP #$3
         BLT CACHECK1
         CMP #$FE
         BGE CACHECK1
FINDA2   INY
         INX
         INX
         INX
         CPX #$2E
         BLT FINDA1
         RTS
CACHECK1 SEC
         LDA $94D5,X
         SBC $94A5,X
         CMP #$1
         BGE CACHECK2
         JMP CACHECK3
CACHECK2 CMP #$FB
         BLT FINDA2
CACHECK3 LDA #$FE
         STA $9473
         LDA $94A3,X
         STA CBYTE1
         STA XBYTE
         LDA $94A4,X
         STA CBIT1
         STA XBIT
         LDA $94A5,X
         STA CY1
         STA Y
         LDA $9400,Y
         STA AYTEMP
         STA SHNUM
         LDA AYTEMP
         CMP #$1
         BEQ C1
         CMP #$4
         BEQ C
         LDA #$C8
         STA $9443,X
         JMP C1.1
C        LDA #$FE
         STA $9443,X
         JMP C1.1
C1       LDA XRIGHT1
         STA $94A3,X
         JSR $8FA2
         JSR MAXSC
         JSR SCORE
         JSR PRINT
         DEC MAX
         LDA MAX
```

DEPTH CHARGE

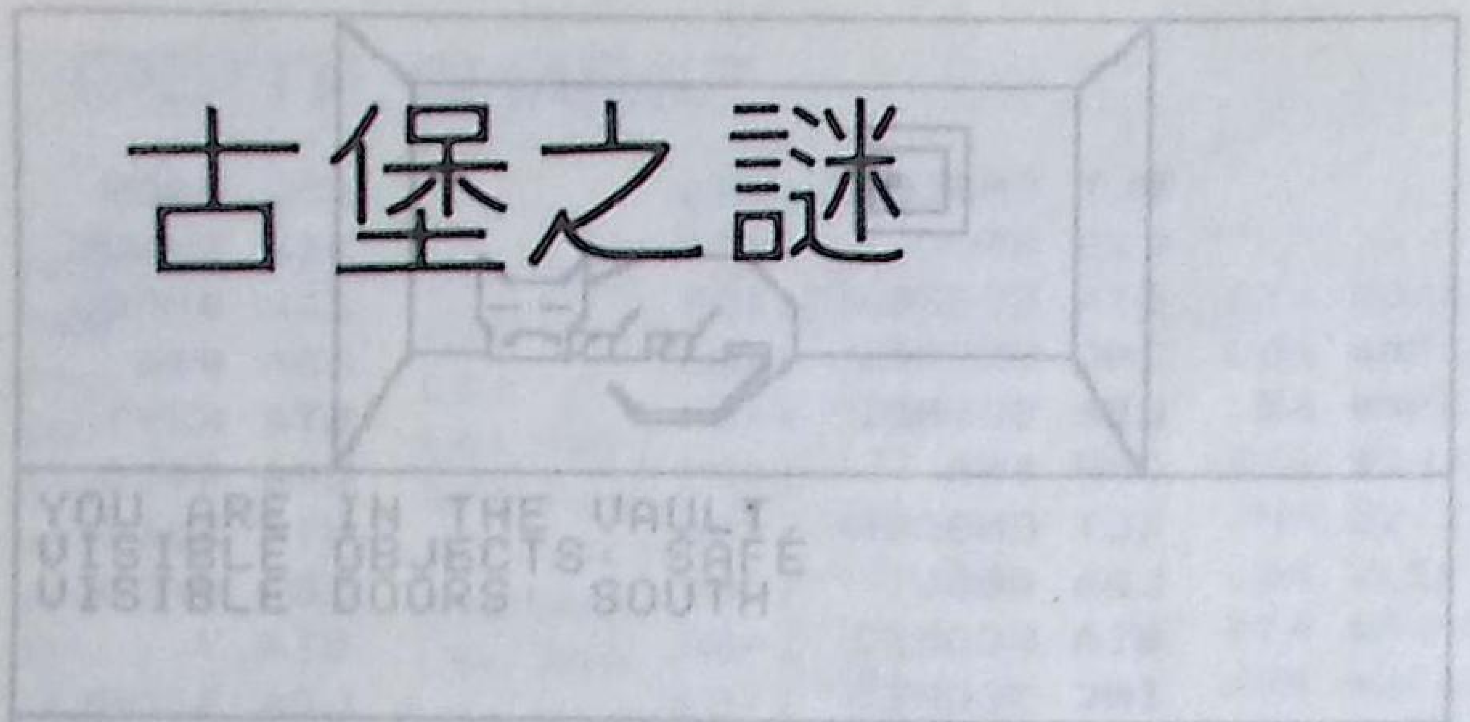


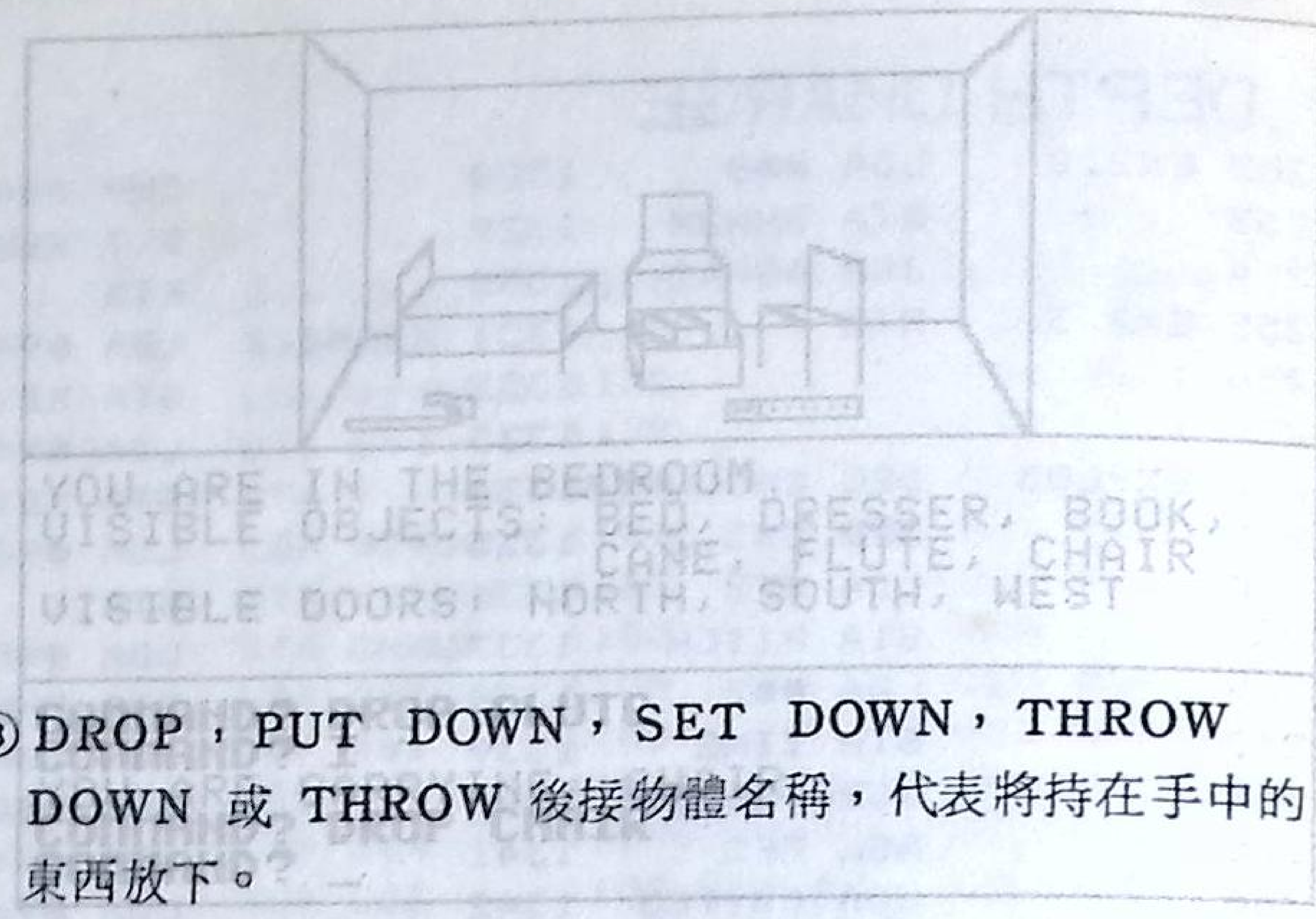
```
         CMP #$6
         BGE C1.2
         LDX XT
         LDA #$FE
         STA $9443,X
         JMP C1.2
C1.1     JSR $8FA2
C1.2     LDA $94D3
         STA XBYTE
         LDA $94D4
         STA XBIT
         LDA $94D5
         STA Y
         LDA #$03
         STA SHNUM
         JSR $8FA2
         LDA #$D0
         STA $94D5
         LDA #$0
         STA CHFLAG
         LDA #$01
         STA CCFLAG1
         LDA #$06
         STA CHCOUNT1
         LDA #$2
         STA $FB
CH1END   RTS
;
CHCL2    LDA CCFLAG2
         BEQ CHCL2.1
         JMP CH2END
CHCL2.1  LDX #$00
         LDY #$1
FINDB1   STX XT
         SEC
         LDA $94D6
         SBC $94A3,X
         CMP #$3
         BLT CBCHECK1
         CMP #$FE
         BGE CBCHECK1
FINDB2   INY
         INX
         INX
         INX
         CPX #$2E
         BLT FINDB1
         RTS
CBCHECK1 SEC
         LDA $94D8
         SBC $94A5,X
         CMP #$1
         BGE CBCHECK2
         JMP CBCHECK3
CBCHECK2 CMP #$FB
         BLT FINDB2
CBCHECK3 LDA #$FE
         STA $9476
         LDA $94A3,X
         STA CBYTE2
         STA XBYTE
         LDA $94A4,X
         STA CBIT2
         STA XBIT
         LDA $94A5,X
         STA CY2
         STA Y
         LDA $9400,Y
         STA BYTEMP
         STA SHNUM
         LDA BYTEMP
         CMP #$1
         BEQ C2
         CMP #$4
         BEQ CC
         LDA #$C8
         STA $94A3,X
         LDA #$FE
CC       LDA #$FE
         STA $9443,X
         JMP C2.2
C2       LDA XLEFT1
         STA $94A3,X
         JSR $8FA2
         JSR SCORE
         JSR MAXSC
         JSR PRINT
         DEC MAX
         LDA MAX
         CMP #$6
         BGE C2.3
         LDX XT
         LDA #$FE
         STA $9443,X
         JMP C2.3
C2.2     JSR $8FA2
C2.3     LDA $94D6
         STA XBYTE
         LDA $94D7
         STA XBIT
         LDA $94D8
         STA Y
         LDA #$03
         STA SHNUM
         JSR $8FA2
         LDA #$D0
         STA $94D8
         LDA #$0
         STA CHFLAG2
         LDA #$01
         STA CCFLAG2
         LDA #$06
         STA CHCOUNT2
         LDA #$02
         STA $FC
CH2END   RTS
;
;
CHCL3    LDA CCFLAG3
         BEQ CHCL3.1
         JMP CH3END
CHCL3.1  LDX #$00
         LDY #$1
FINDC1   STX XT
         SEC
         LDA $94D9
         SBC $94A3,X
         CMP #$3
         BLT CCCHECK1
         CMP #$FE
         BGE CCCHECK1
FINDC2   INY
         INX
         INX
         INX
         CPX #$2E
         BLT FINDC1
         RTS
CCCHECK1 SEC
         LDA $94DB
         SBC $94A5,X
         CMP #$1
         BGE CCCHECK2
         JMP CCCHECK3
CCCHECK2 CMP #$FB
         BLT FINDC2
CCCHECK3 LDA #$FE
         STA $9479
         LDA $94A3,X
         STA CBYTE3
         STA XBYTE
         LDA $94A4,X
         STA CBIT3
         STA XBIT
         LDA $94A5,X
         STA CY3
         STA Y
         LDA $9400,Y
         STA CYTEMP
         STA SHNUM
         LDA CYTEMP
         CMP #$1
         BEQ C3
         CMP #$4
         BEQ CCC
         LDA #$C8
         STA $94A3,X
         JMP C3.3
CCC      LDA #$FE
         STA $9443,X
         JMP C3.3
C3       LDA XRIGHT1
         STA $94A3,X
         JSR $8FA2
         JSR SCORE
         JSR MAXSC
         JSR PRINT
         DEC MAX
         LDA MAX
         CMP #$6
         BGE C3.4
         LDX XT
         LDA #$FE
         STA $9443,X
         JMP C3.4
C3.3     JSR $8FA2
C3.4     LDA $94D9
         STA XBYTE
         LDA $94DA
         STA XBIT
         LDA $94DB
         STA Y
         LDA #$03
         STA SHNUM
         JSR $8FA2
         LDA #$D0
         STA $94DB
         LDA #$0
         STA CHFLAG3
         LDA #$01
         STA CCFLAG3
         LDA #$06
         STA CHCOUNT3
         LDA #$02
         STA $FD
CH3END   RTS
;
FSOUND   LDA $C030
S1       DEY
         BNE S2
         DEC TIME
         BEQ SEND
S2       DEX
         BNE S1
         LDX PITCH
         JMP FSOUND
SEND     RTS
;
;
EXPLO1   DEC CHCOUNT1
         BEQ EX1.2
         LDA $FB
         STA PITCH
         LDA #$02
         STA TIME
         JSR FSOUND
         ASL $FB
         LDA CBYTE1
         STA XBYTE
         LDA CBIT1
         STA XBIT
         LDA CY1
         STA Y
         LDA AYTEMP
         CMP #$1
         BEQ EX1.A
         LDA #$6
         STA SHNUM
         JSR $8F00
         JMP EX1.3
EX1.A    LDA #$8
         STA SHNUM
         JSR $8F00
         JMP EX1.3
EX1.2    LDA #$00
         STA CCFLAG1
         LDA CBYTE1
         STA XBYTE
         LDA CBIT1
         STA XBIT
         LDA CY1
         STA Y
         LDA AYTEMP
         CMP #$1
         BEQ EX1.B
         LDA #$6
         STA SHNUM
         JSR $8FA2
         RTS
EX1.B    LDA #$8
         STA SHNUM
         JSR $8FA2
EX1.3    RTS
;
EXPLO2   DEC CHCOUNT2
         BEQ EX2.2
         LDA $FC
         STA PITCH
         LDA #$2
         STA TIME
         JSR FSOUND
         ASL $FC
         LDA CBYTE2
         STA XBYTE
         LDA CBIT2
         STA XBIT
         LDA CY2
         STA Y
         LDA BYTEMP
         CMP #$1
         BEQ EX2.A
         LDA #$6
         STA SHNUM
         JSR $8F00
         JMP EX2.3
EX2.A    LDA #$8
         STA SHNUM
         JSR $8F00
         JMP EX2.3
EX2.2    LDA #$00
         STA CCFLAG2
         LDA CBYTE2
         STA XBYTE
         LDA CBIT2
         STA XBIT
         LDA CY2
         STA Y
         LDA BYTEMP
         CMP #$1
         BEQ EX2.B
         LDA #$6
         STA SHNUM
         JSR $8FA2
         RTS
```

DEPTH CHARGE

```
EX2.B    LDA #$8
         STA SHNUM
         JSR $8FA2
EX2.3    RTS
;
;
EXPLO3   DEC CHCOUNT3
         BEQ EX3.2
         LDA $FD
         STA PITCH
         LDA #$2
         STA TIME
         JSR FSOUND
         ASL $FD
         LDA CBYTE3
         STA XBYTE
         LDA CBIT3
         STA XBIT
         LDA CY3
         STA Y
         LDA CYTEMP
         CMP #$1
         BEQ EX3.A
         LDA #$6
         STA SHNUM
         JSR $8F00
         JMP EX3.3
EX3.A    LDA #$8
         STA SHNUM
         JSR $8F00
         JMP EX3.3
EX3.2    LDA #$00
         STA CCFLAG3
         LDA CBYTE3
         STA XBYTE
         LDA CBIT3
         STA XBIT
         LDA CY3
         STA Y
         LDA CYTEMP
         CMP #$1
         BEQ EX3.B
         LDA #$6
         STA SHNUM
         JSR $8FA2
         RTS
EX3.B    LDA #$8
         STA SHNUM
         JSR $8FA2
EX3.3    RTS
;
;
;
XBOMB1   LDA $94CC
         CMP #$40
         BLT XBOMB.1
         RTS
XBOMB.1  LDA $94CA
         STA XBYTE
         LDA $94CB
         STA XBIT
         LDA $94CC
         STA Y
         LDA #$5
         STA SHNUM
         JSR $8FA2
         LDA #$BF
         STA $94CC
         LDA $94A0
         CLC
         ADC XDIST1
         STA $94CA
         RTS
;
;
XBOMB2   LDA $94CF
         CMP #$40
         BLT XBOMB.2
         RTS
XBOMB.2  LDA $94CD
         STA XBYTE
         LDA $94CE
         STA XBIT
         LDA $94CF
         STA Y
         LDA #$5
         STA SHNUM
         JSR $8FA2
         LDA #$BF
         STA $94CF
         LDA $94A0
         CLC
         ADC XDIST2
         STA $94CD
         RTS
;
;
XBOMB3   LDA $94D2
         CMP #$40
         BLT XBOMB.3
         RTS
XBOMB.3  LDA $94D0
         STA XBYTE
         LDA $94D1
         STA XBIT
         LDA $94D2
         STA Y
         LDA #$5
         STA SHNUM
         JSR $8FA2
         LDA #$BF
         STA $94D2
         LDA $94A0
         SEC
         SBC XDIST3
         STA $94D0
         RTS
;
;
XBOMB    LDX #$00
;
;
;
XBCHK    LDX #$00
XBCHK1   LDA $94C7,X
         CMP #$29
         BEQ XBCHK3
         CMP #$FB
         BEQ XBCHK4
XBCHK2   INX
         INX
         INX
         CPX #$A
         BLT XBCHK1
         RTS
XBCHK3   LDA #$FC
         STA $94C7,X
         JMP XBCHK2
XBCHK4   LDA #$28
         STA $94C7,X
         JMP XBCHK2
;
;
SCORE    INC SCORE0
         LDA SCORE0
         CMP #$A
         BLT ENSCORE
         LDA #$00
         STA SCORE0
SCORE.1  INC SCORE1
         LDA SCORE1
         CMP #$A
         BLT ENSCORE
         LDA #$00
         STA SCORE1
         INC SCORE2
         LDA SCORE2
         CMP #$A
         BLT ENSCORE
         LDA #$00
         STA SCORE2
         INC SCORE3
         LDA SCORE3
         CMP #$A
         BLT ENSCORE
         LDA #$00
         STA SCORE3
         INC SCORE4
         LDA SCORE4
         CMP #$A
         BLT ENSCORE
         LDA #$00
         STA SCORE0
         STA SCORE1
         STA SCORE2
         STA SCORE3
         STA SCORE4
ENSCORE  RTS
;
;
;
MAXSC    DEC MAXSC1
         LDA MAXSC1
         CMP #$FF
         BNE MAXEND
         LDA #$9
         STA MAXSC1
         DEC MAXSC2
         LDA MAXSC2
         CMP #$FF
         BNE MAXEND
         LDA #$00
         STA MAXSC1
         STA MAXSC2
MAXEND   RTS
;
;
;
PRINT    LDA #$3
         STA XBYTE
         LDA #$00
         STA XBIT
         LDA #$00
         STA Y
         LDA SCORE4
         CLC
         ADC #$09
         STA SHNUM
         JSR $8F00
         LDA #$4
         STA XBYTE
         LDA #$00
         STA XBIT
         LDA #$00
         STA Y
         LDA SCORE3
         CLC
         ADC #$09
         STA SHNUM
         JSR $8F00
         LDA #$5
         STA XBYTE
         LDA #$00
         STA XBIT
         LDA #$00
         STA Y
         LDA SCORE2
         CLC
         ADC #$09
         STA SHNUM
         JSR $8F00
         LDA #$6
         STA XBYTE
         LDA #$00
         STA XBIT
         LDA #$00
         STA Y
         LDA SCORE1
         CLC
         ADC #$09
         STA SHNUM
         JSR $8F00
         LDA #$7
         STA XBYTE
         LDA #$00
         STA XBIT
         LDA #$00
         STA Y
         LDA SCORE0
         CLC
         ADC #$09
         STA SHNUM
         JSR $8F00
         LDA #$8
         STA XBYTE
         LDA #$0
         STA XBIT
         LDA #$0
         STA Y
         LDA #$9
         STA SHNUM
         JSR $8F00
         LDA #$1C
         STA XBYTE
         LDA #$0
         STA XBIT
         LDA #$0
         STA Y
         LDA MAXSC2
         CLC
         ADC #$9
         STA SHNUM
         JSR $8F00
         LDA #$1D
         STA XBYTE
         LDA #$0
         STA XBIT
         LDA #$0
         STA Y
         LDA MAXSC1
         CLC
         ADC #$9
         STA SHNUM
         JSR $8F00
         RTS
;
;
;
DELAY    SEC
         LDA NUMBER
         SBC MAX
P4       BEQ P6
         LDX #$FE
P8       DEX
         BNE P8
         SEC
         SBC #$01
         JMP P4
P6       RTS
;
;
;
;
         END
```

古堡之謎





十七世紀法國大革命令很多貴族喪生在斷頭台，不過有些機警的貴族還是逃之夭夭，匿居在其安樂窩上安然渡餘年，其中一位就是伯尼兹公爵，傳說他擁有龐大的財產，大部份都收藏在城堡之中，這位公爵並沒有承繼人，他死後，遺下的財實便藏在古堡中。因此引起不少好奇之士，但却沒有人能發現藏實所在。假如你是第一位發現這個秘密的人便可以得到巨額財實了！

閣下進入本遊戲後，可以在每間房內尋找到藏實地點的綫索，不過要小心，可能發生的危險意外，要成功取得實藏需要依賴常識，推理思巧及百折不撓的精神。

本遊戲是一個 ADVENTURE GAME 程式執行後，便會出現一個高解像度圖形，按入任何鍵便出現指示文字，這些指示可分爲三部份，最上端的部份是你所看見環繞四周的環境，中間是解釋的文字敍述，每次你改變方向圖環繞形都會跟着改變。

閣下可使用簡單的英語來告訴電影你的每一步行動，例如〔GO NORTH〕代表前往北面，這些鍵入的命令將在螢幕的下端顯示出來，緊隨在 COMMAND？之後。通常情形下所鍵入的命令必須是動詞爲首名詞隨後的，長度爲一個字至4個字不等，一切的標點符號皆不需使用，表（一）列出一般的指令及他們的意義。

本遊戲的玩法

GET〔物體〕：檢取一般物件往手上，每次最多只可使用兩次，每隻手可持一件。

DROP〔物件〕：將物件放在地上或放在有空間的地方。

GO〔方向〕：按所鍵入的方向穿過門。

INVENTORY：列出所持在手上的物件。

LOOK〔物體〕：詳細視察一件物件。

SAVE GAME：產生一個名爲 GAMEFILE 的檔案在磁碟上，包括此遊戲的資料以代下次繼續。

LOAD GAME：將磁碟上的 GAMEFILE 取出繼續上次未完的遊戲。

END GAME：離開本遊戲回到 BASIC 上。

RESTART GAME：重新開始本遊戲。

打入程式需注意事項

①ENTER，GO TO，GO INSIDE，GO INTO，GO IN，GO，CLIMB ONTO，CLIMB ON，CLIMB，STEP ONTO，STEP ON 或 STEP 都代表有相同意義，他們可後接方向，或其他欲前往的地方。

②GET，TAKE，PICK UP或GRAB 後接物體，表示檢在手中。

③DROP，PUT DOWN，SET DOWN，THROW DOWN 或 THROW 後接物體名稱，代表將持在手中的東西放下。

④OPEN，PRY OPEN，PRY，PULL OPEN 或 PULL 後緊接物體名稱代表開始該物體。

⑤LOOK AT，LOOK INTO或LOOK IN，LOOK，EXAMIME 或 SEARCH 後緊接物件名稱，代表檢查細看該物件。

⑥LOOK，HEAT 或 BOIL 後接可食用的東西名稱。

⑦TURN ON，SWITCH ON，ACTIVATE，LIG－HT UP，LIGHT 或 FIRE UP，FIRE 或 IGNITE 照明物件或可燃燒的東西。

⑧TURN，OF，SWITCH OFF，SHUT OFF或 DEACTIVATE 照明物體或可燃燒的物體。

⑨PLAY，BLOW INTO，BLOW 樂器。

⑩DIAL、TURN TO，COMBINATION，SPIN COMBINATION，SPIN TO，SPIN DIAL TO，SPIN KNOB TO，TURN DIAL TO，TURN KNOB TO，POSITION KNOB ON 或 POSITION DIAL AT 後接保險箱數字。

⑪WEAR，PUT ON，WRAP ON或WRAP 後接物佈名稱，在寒冷的環境下會很有幫助。

⑫READ 物件上寫的東西。

⑬EAT或DRINK 或 CONSUME 可供食用的東西名稱。

⑮KILL 或HIT或DESTROY對你有害的動物，不過要小心，切勿不自量力。

⑭CLOSE 或 SHUT 一些物件如門等。

⑯QUIT 或 END或 LEAVE 或 STOP可中止該遊戲。

上述只是本遊戲所採用的指令的一部份舉例，讀者們必須留意。

大致上，這些命令都很簡明，故不疑特別討論，不過大家要留意，每次遊戲開始都會將實藏及其它資料以隨機方式安放，因此絕難相同。

其中一些指令是可以簡化其輸入形式的，例如 GO N－ORTH 可簡化爲 GO N，INUENTORY 可簡化爲I，但其他命令必須全部鍵入。一旦電腦對你所按入的指令不解時會出現一個錯誤的訊息，祝各位好運！有時一局遊戲可能要玩很久，閣下可以不妨在中途將O遊戲 SAVE 下來（由 SAVE GAME 指令），留待下次有空是繼續完成它。

其他有效指令如下：

此遊戲的長度超過16K，在設計時爲了使字體美麗，故此採用了 APPLESOFT TOOL KIT 內的文字造形，

古堡之謎

採用重定位的技術將它們載入往緊隨 DOS 的指標之下，再重置 HIMEM 數值，因此若閣下預備自己鍵入本程式時，本程式共有 2 個列表： ADVENTURE CASTLEB CA—STLE）必須擁有 APPLESOFT TOOL KIT 磁碟，將其中：

（1） HRCG
（2） RLOAD
（3） RBOOT
（4） FLOW.SET

```
10  REM     ***********************
20  REM     *        CASTLE       *
30  REM     *    BY COMPUHOUSE    *
40  REM     * COPYRIGHT (C) 1984 *
70  REM     ***********************
80 :
90  GOTO 2740
100  GOSUB 120: IF F = 0 THEN  POP
     : PRINT "YOU DON'T HAVE ";QT
     $;N$;QT$;".": GOTO 1030
110  RETURN
120 F = 0: IF NP = 0 THEN  RETURN
130  FOR XX = 1 TO NP: IF P$(XX) =
     N$ THEN F = XX:XX = NP
140  NEXT XX: RETURN
150  GOSUB 170: IF F = 0 THEN  POP
     : PRINT QT$;N$;QT$;" ISN'T H
     ERE.": GOTO 1030
160  RETURN
170 F = 0: IF NO(RM) = 0 THEN  RETURN
180  FOR XX = 1 TO NO(RM): IF O$(
     RM,XX) = N$ THEN F = XX:XX =
     NO(RM)
190  NEXT XX: RETURN
200  GOSUB 120: IF F = 0 THEN  GOSUB
     150
210  RETURN
220  GOSUB 120: IF F = 0 THEN  GOSUB
     170
230  RETURN
240 DI$ =  MID$ (DE$,(D - 1) * 5 +
     1,5 - (D = 2 OR D = 4)): RETURN
250 H9 =  PEEK (36):V9 =  PEEK (3
     7): RETURN
260  PRINT "O.K.": GOTO 1030
270  PRINT CY$;: VTAB 2: HTAB 2: PRINT
     CV$;: VTAB 11: HTAB 38: PRINT
     CW$;CP$;
280  HPLOT 63,4 TO 63,91 TO 77,68
     TO 77,16 TO 63,4: HPLOT 77,
     16 TO 203,16 TO 217,4 TO 217
     ,91: HPLOT 203,16 TO 203,68 TO
     217,91: HPLOT 203,68 TO 77,6
     8
290  IF RM <  > 5 THEN 340
```



（5） GOTHIC.SET

5 個檔案抄往同一磁碟上，本刊出版的即期程式磁碟則包括上述五個檔案。

第一個 ADVENTURE CASTLE 程式是一個轉載程式，它啓動高解像圖形文字產生器及載入文字造形，其次會以 TEXT 指令執行一個載入主程式 CASTLE 的 TEXT FILE 。使主程式載入到高解像圖形之上並執行它，閣下無須自己建立此 TEXT FILE ，因爲在第一次執行 AD—VENTURN CASTLE 時會自動建立。□

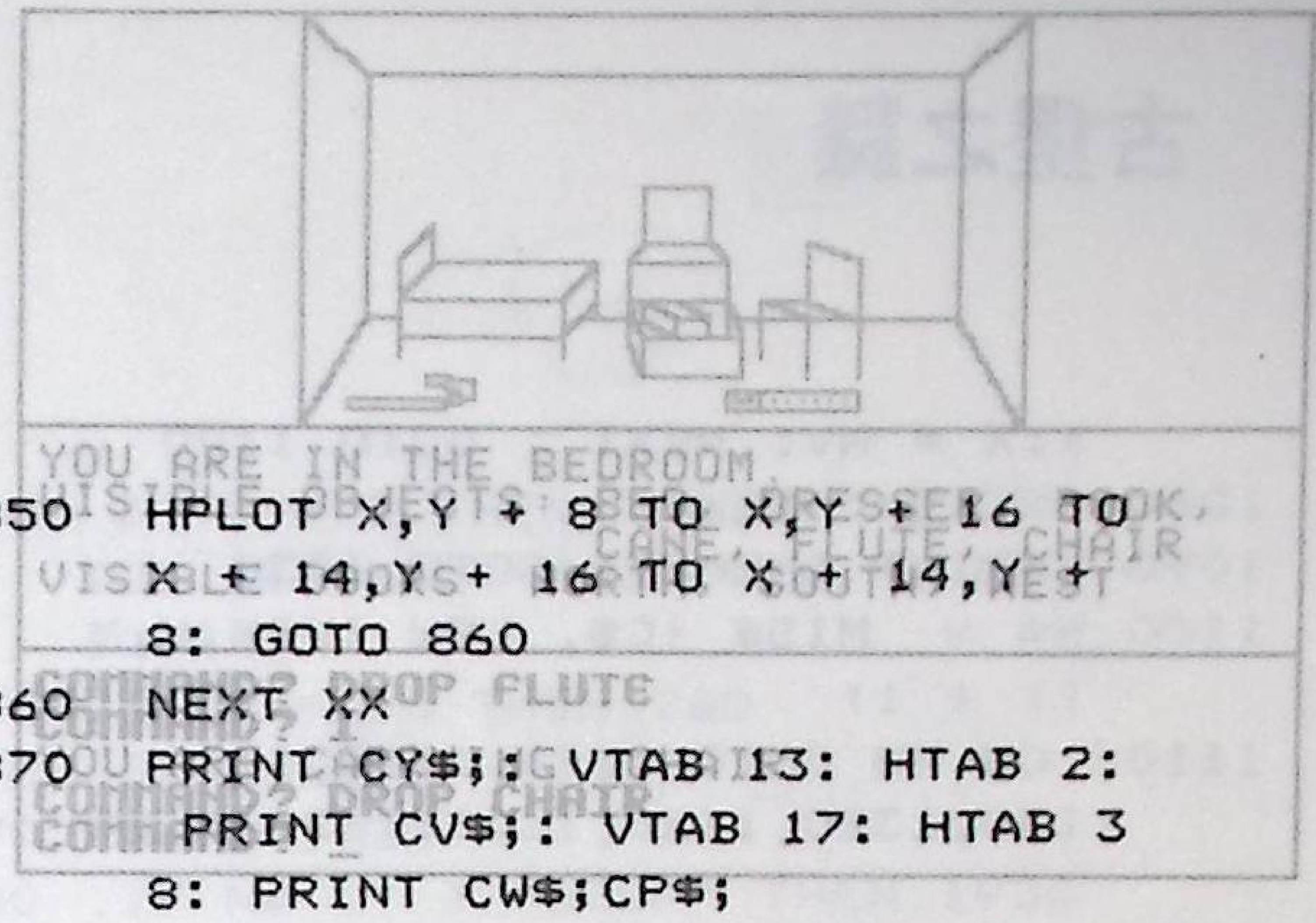
```
300  HCOLOR= 0: HPLOT 108,68 TO 8
     4,68: HCOLOR= 3: HPLOT 108,4
     0 TO 108,68 TO 105,72 TO 105
     ,52 TO 108,48 TO 87,48 TO 87
     ,40 TO 108,40
310  HPLOT 87,48 TO 84,52 TO 105,
     52: HPLOT 84,52 TO 84,72 TO
     105,72: HPLOT 87,60 TO 101,6
     0 TO 101,64 TO 87,64 TO 87,6
     0
320  HCOLOR= 0: HPLOT 143,68 TO 1
     19,68: HCOLOR= 3: HPLOT 119,
     32 TO 140,32 TO 140,72 TO 11
     9,72 TO 119,32: IF IC = 0 THEN
     HPLOT 122,44 TO 122,60: HPLOT
     140,32 TO 143,32 TO 143,68 TO
     140,72: GOTO 650
330  HPLOT 122,36 TO 136,36 TO 13
     6,68 TO 122,68 TO 122,36: HCOLOR=
     0: HPLOT 143,68 TO 154,68: HCOLOR=
     3: HPLOT 140,32 TO 154,32 TO 154,
     76 TO 140,72: GOTO 650
340  IF RM <  > 8 THEN 400
350  HCOLOR= 0: HPLOT 84,68 TO 12
     2,68: HCOLOR= 3: HPLOT 84,76
     TO 84,56 TO 91,48 TO 91,56 TO
     84,64 TO 119,64 TO 126,56 TO
     91,56
360  HPLOT 84,72 TO 119,72 TO 126
     ,64: HPLOT 119,64 TO 119,76:
     HPLOT 126,56 TO 126,68: HCOLOR=
     0: HPLOT 133,68 TO 154,68: HCOLOR= 3
370  HPLOT 133,72 TO 133,56 TO 15
     4,56 TO 154,64 TO 133,64: HPLOT
     133,56 TO 136,52 TO 150,52 TO
     154,56
380  HPLOT 136,52 TO 136,40 TO 15
     0,40 TO 150,52: IF DR = 0 THEN
     HPLOT 133,72 TO 154,72 TO 1
     54,64: GOTO 650
390  HCOLOR= 0: HPLOT 154,68 TO 1
     56,68: HCOLOR= 3: HPLOT 133,
     64 TO 136,72 TO 157,72 TO 15
     4,64 TO 154,72: HPLOT 133,72
     TO 136,80 TO 136,72: HPLOT
     136,80 TO 157,80 TO 157,72: GOTO
     650
400  IF RM <  > 4 THEN 630
410  HPLOT 161,24 TO 182,24 TO 18
     2,44 TO 161,44 TO 161,24: HPLOT
     164,28 TO 178,28 TO 178,40 TO
     164,40 TO 164,28
```

古堡之謎

```
420  HPLOT 167,33 TO 169,33 TO 16
     9,35 TO 167,35 TO 167,33
430  IF BE = 0 THEN 520
440 X = 90:Y = 49: HCOLOR= 0: HPLOT
     X + 2,Y + 19 TO X + 17,Y + 1
     9: HPLOT X + 20,Y + 19 TO X +
     32,Y + 19: HPLOT X + 35,Y +
     19 TO X + 47,Y + 19: HPLOT X
      + 55,Y + 19 TO X + 60,Y + 1
     9: HCOLOR= 3
450  HPLOT X,Y TO X + 5,Y + 5 TO
     X + 15,Y + 5 TO X + 20,Y TO
     X + 20,Y + 12 TO X + 17,Y +
     15 TO X + 17,Y + 20 TO X + 2
     ,Y + 20 TO X + 2,Y + 15 TO X
     ,Y + 12 TO X,Y
460  HPLOT X + 20,Y + 5 TO X + 30
     ,Y TO X + 50,Y TO X + 60,Y +
     10 TO X + 60,Y + 20 TO X + 4
     5,Y + 32 TO X + 30,Y + 32 TO
     X + 25,Y + 25 TO X + 30,Y +
     30 TO X + 45,Y + 30 TO X + 5
     5,Y + 20
470  HPLOT  TO X + 55,Y + 17 TO X
      + 47,Y + 17 TO X + 47,Y + 2
     0 TO X + 40,Y + 20 TO X + 40
     ,Y + 17 TO X + 45,Y + 10
480  HPLOT X + 40,Y + 20 TO X + 3
     5,Y + 20 TO X + 35,Y + 17 TO
     X + 37,Y + 15: HPLOT X + 41,
     Y + 15 TO X + 32,Y + 15 TO X
      + 32,Y + 20 TO X + 25,Y + 2
     0 TO X + 25,Y + 17 TO X + 30
     ,Y + 10
490  HPLOT X + 25,Y + 20 TO X + 2
     0,Y + 20 TO X + 20,Y + 17 TO
     X + 22,Y + 15: HPLOT X + 17,
     Y + 15 TO X + 26,Y + 15: HPLOT
     X + 2,Y + 10 TO X + 7,Y + 10
500  HPLOT X + 2,Y + 10 TO X + 7,
     Y + 10: HPLOT X + 12,Y + 10 TO
     X + 17,Y + 10: HPLOT X + 5,Y
      + 17 TO X + 7,Y + 15 TO X +
     12,Y + 15 TO X + 15,Y + 17
510  GOTO 650
520 X = 86:Y = 34: HCOLOR= 0: HPLOT
     X + 47,Y + 34 TO X + 50,Y +
     34: HPLOT X + 15,Y + 34: HPLOT
     X + 20,Y + 34: HPLOT X + 27,
     Y + 34: HPLOT X + 35,Y + 34:
      HPLOT X + 37,Y + 34: HPLOT
     X + 40,Y + 34: HCOLOR= 3
530  HPLOT X,Y + 10 TO X + 5,Y +
     15 TO X + 15,Y + 15 TO X + 2
     0,Y + 10 TO X + 20,Y + 22 TO
     X + 17,Y + 25 TO X + 17,Y +
     30 TO X + 2,Y + 30 TO X + 2,
     Y + 25 TO X,Y + 22 TO X,Y +
     10
540  HPLOT X + 20,Y + 15 TO X + 2
     7,Y + 10 TO X + 40,Y + 10 TO
     X + 50,Y + 15 TO X + 55,Y +
     25 TO X + 65,Y + 25 TO X + 7
     0,Y + 20 TO X + 70,Y + 10 TO
     X + 65,Y + 5 TO X + 60,Y + 5
550  HPLOT  TO X + 55,Y + 10 TO X
      + 57,Y + 2 TO X + 67,Y + 2 TO
     X + 72,Y + 10 TO X + 72,Y +
     20 TO X + 65,Y + 27 TO X + 5
     0,Y + 27 TO X + 50,Y + 35 TO
     X + 47,Y + 35 TO X + 45,Y +
     32
560  HPLOT  TO X + 40,Y + 35 TO X
      + 40,Y + 33 TO X + 45,Y + 3
     0 TO X + 40,Y + 25 TO X + 32
     ,Y + 25 TO X + 32,Y + 30 TO
     X + 35,Y + 35 TO X + 30,Y +
     32 TO X + 27,Y + 35 TO X + 2
     5,Y + 32 TO X + 20,Y + 35 TO
     X + 25,Y + 30
570  HPLOT  TO X + 30,Y + 30 TO X
      + 30,Y + 25 TO X + 17,Y + 2
     5: HPLOT X + 22,Y + 25 TO X +
     22,Y + 30 TO X + 20,Y + 30 TO
     X + 15,Y + 35 TO X + 20,Y +
     32 TO X + 22,Y + 32
580  HPLOT X + 37,Y + 25 TO X + 4
     2,Y + 30 TO X + 37,Y + 32 TO
     X + 37,Y + 35 TO X + 40,Y +
     35
590  HPLOT X + 3,Y + 17 TO X + 7,
     Y + 17 TO X + 7,Y + 19 TO X +
     5,Y + 19 TO X + 5,Y + 17: HPLOT
     X + 17,Y + 17 TO X + 13,Y +
     17 TO X + 13,Y + 19 TO X + 1
     5,Y + 19 TO X + 15,Y + 17
600  HPLOT X + 7,Y + 25 TO X + 5,
     Y + 27 TO X + 5,Y + 25 TO X +
     7,Y + 22 TO X + 12,Y + 22 TO
     X + 15,Y + 25 TO X + 15,Y +
     27 TO X + 12,Y + 25: HPLOT X
      + 8,Y + 22 TO X + 8,Y + 27
610  HPLOT X + 12,Y + 22 TO X + 1
     2,Y + 27: HPLOT X + 9,Y + 25
      TO X + 11,Y + 25: HPLOT X +
     25,Y + 30 TO X + 25,Y + 25
620  GOTO 650
630  IF RM <  > 9 THEN 650
640  HPLOT 112,80 TO 112,64 TO 11
     9,56 TO 147,56 TO 147,72: HPLOT
```

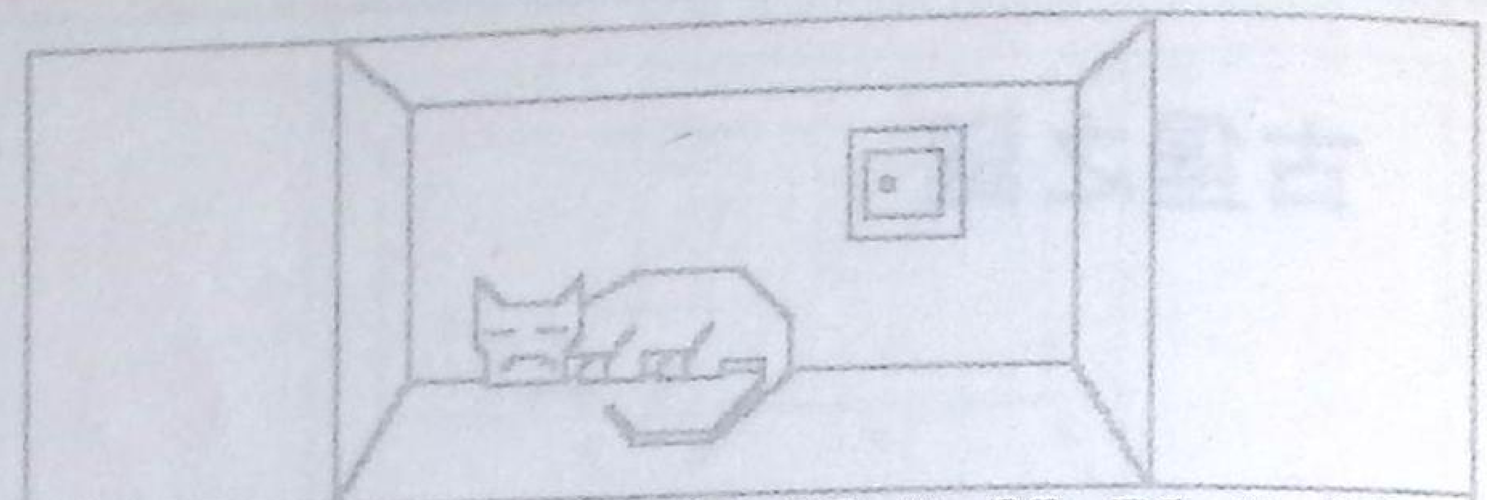
古堡之謎

```
   147,56 TO 140,64 TO 140,80: HPLOT
   140,64 TO 112,64: HPLOT 119,
   64 TO 119,72: GOTO 650
650  IF NO(RM) = 0 THEN 870
660  FOR XX = 1 TO NO(RM):N$ = O$
   (RM,XX)
670  IF N$ < > "CANE" THEN 690
680  HPLOT 73,87 TO 73,85 TO 94,8
   5 TO 97,83 TO 94,82 TO 90,82
    TO 90,80 TO 100,82 TO 100,8
   5 TO 94,87 TO 73,87: GOTO 86
   0
690  IF N$ < > "RUG" THEN 710
700  HPLOT 125,82 TO 153,82 TO 14
   7,87 TO 119,87 TO 125,82: GOTO
   860
710  IF N$ < > "CHAIR" THEN 730
720  HPLOT 161,76 TO 161,64 TO 17
   1,64 TO 171,52 TO 182,56 TO
   182,68 TO 171,64: HPLOT 161,
   64 TO 171,68 TO 171,80: HPLOT
   182,68 TO 182,80: GOTO 860
730  IF N$ < > "BOOK" THEN 790
740  IF RM = 5 THEN X = 189:Y = 7
   6: GOTO 780
750  IF RM = 8 AND BP = 0 THEN X =
   136:Y = 64: GOTO 780
760  IF RM = 4 OR RM = 8 THEN X =
   103:Y = 79: GOTO 780
770 X = 91:Y = 70
780  HPLOT X,Y TO X + 7,Y TO X +
   14,Y + 4 TO X + 7,Y + 4 TO X
    + 7,Y + 8 TO X,Y + 4 TO X,Y
   : HPLOT X + 7,Y + 8 TO X + 1
   4,Y + 8 TO X + 14,Y + 4: HPLOT
   X,Y TO X + 7,Y + 4: GOTO 860
790  IF N$ < > "FLUTE" THEN 820
800  HPLOT 154,83 TO 182,83 TO 18
   2,87 TO 154,87 TO 154,83: HPLOT
   156,83 TO 156,85 TO 159,85 TO
   159,83: HPLOT 161,83 TO 161,
   87
810  FOR X = 164 TO 179 STEP 3: HPLOT
   X,85: NEXT : GOTO 860
820  IF N$ < > "KETTLE" THEN 860
830 X = 189:Y = 71: IF RM = 5 THEN
   X = 90:Y = 35: HCOLOR= 0: HPLOT
   X,Y + 13 TO X + 14,Y + 13: HPLOT
   X + 3,Y + 5 TO X + 10,Y + 5:
    HCOLOR= 3
840  HPLOT X + 3,Y + 4 TO X,Y TO
   X + 14,Y TO X + 10,Y + 4 TO
   X + 3,Y + 4 TO X,Y + 8 TO X +
   14,Y + 8 TO X + 10,Y + 4
```




```
850  HPLOT X,Y + 8 TO X,Y + 16 TO
   X + 14,Y + 16 TO X + 14,Y +
   8: GOTO 860
860  NEXT XX
870  PRINT CY$;: VTAB 13: HTAB 2:
    PRINT CV$;: VTAB 17: HTAB 3
   8: PRINT CW$;CP$;
880  PRINT "YOU ARE IN THE ";N$(R
   M);".": PRINT "VISIBLE OBJEC
   TS: ";: IF NO(RM) = 0 THEN  PRINT
   "NONE": GOTO 980
890  FOR X = 1 TO NO(RM):Q9 =  LEN
   (O$(RM,X)): IF X < NO(RM) THEN
   Q9 = Q9 + 2
900 Q8 =  PEEK (36) + Q9 -  PEEK
   (33): IF Q8 < = 0 THEN 940
910  IF X = NO(RM) THEN 930
920  Q8 = Q8 - 1: IF Q8 = 0 THEN  PRINT
   O$(RM,X);",";: HTAB 18: GOTO
   970
930  PRINT : HTAB 18: GOTO 950
940  IF Q8 = 0 AND X < NO(RM) THEN
    PRINT O$(RM,X);",": HTAB 18
   : GOTO 970
950  PRINT O$(RM,X);: IF X < NO(R
   M) THEN  PRINT ", ";: GOTO 9
   70
960  IF Q8 = 0 THEN  NEXT : GOTO
   980
970  NEXT X: PRINT
980  PRINT "VISIBLE DOORS: ";:DF =
   0: FOR D = 1 TO 4: IF R(RM,D
   ) = 0 THEN 1010
990  IF DF THEN  PRINT ", ";
1000 DF = 1: GOSUB 240: PRINT DI$
   ;
1010  NEXT D
1020  PRINT CY$;: VTAB 19: HTAB 2
   : PRINT CV$;: VTAB 23: HTAB
   38: PRINT CW$;: VTAB V9 + 1:
   HTAB H9 + 1
1030  PRINT CA$;"1";: INPUT "COMM
   AND? ";C$: PRINT CA$;"0";: GOSUB
   250
1040  IF  RIGHT$ (C$,1) = " " THEN
   C$ =  LEFT$ (C$, LEN (C$) -
   1): GOTO 1040
1050  IF C$ = "N" OR C$ = "E" OR
   C$ = "S" OR C$ = "W" THEN C$
   = "GO " + C$
1060  IF C$ = "DESCEND FROM CHAIR
   " THEN C$ = "GO DOWN"
1070  FOR X = 1 TO NV: FOR YY = 1
   TO NS(X): IF VB$(X,YY) =  LEFT$
   (C$, LEN (VB$(X,YY))) THEN Y
   = YY:YY = NS(X): NEXT :V =
```

古堡之謎



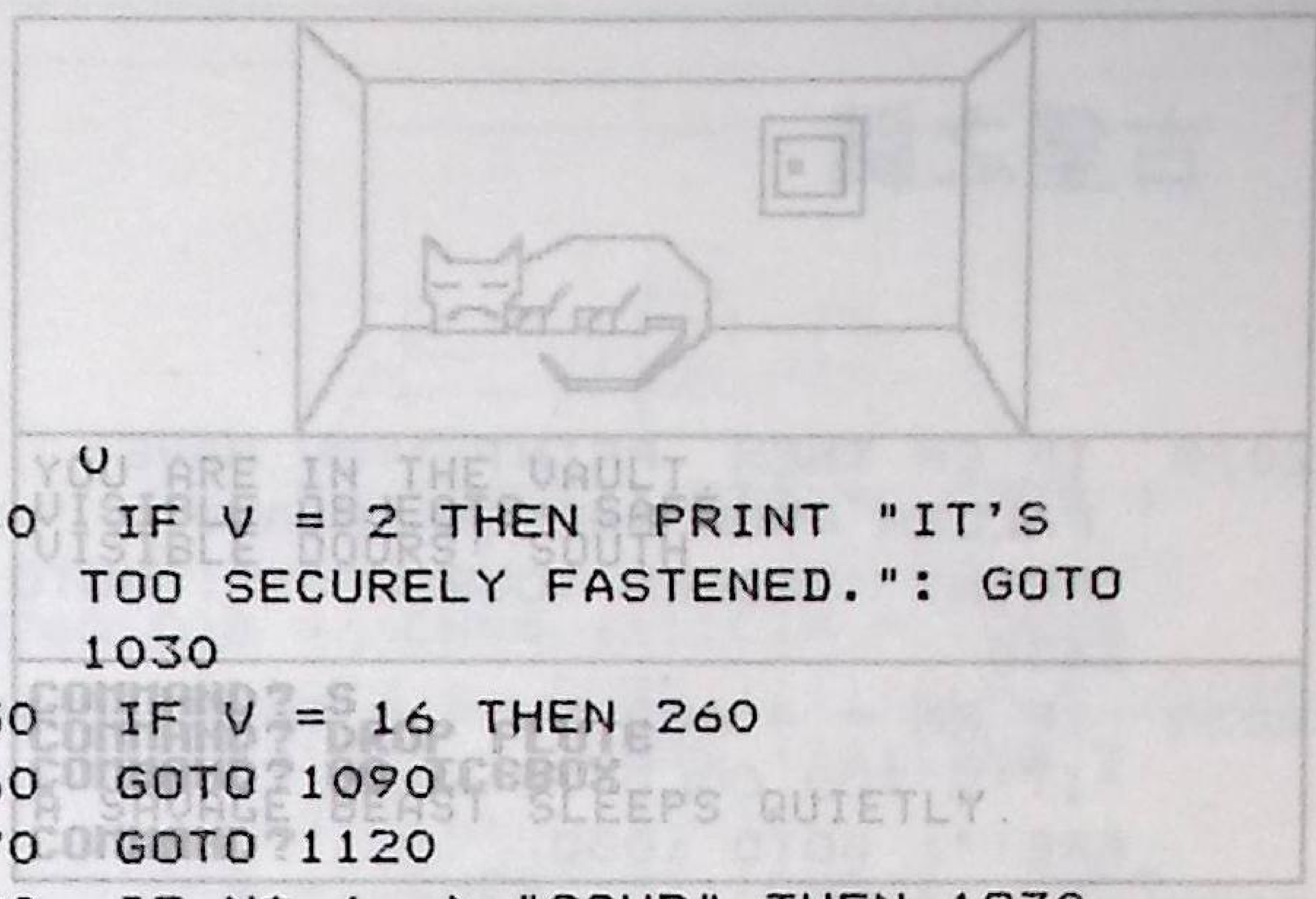
```
    X:X = NV: NEXT : GOTO 1100
1080  NEXT : NEXT
1090  PRINT "HUH?": GOTO 1030
1100 N$ =  MID$ (C$, LEN (VB$(V,Y
    )) + 1)
1110  ON RM GOTO 1120,1120,1120,1
    130,1320,1120,1120,1620,1120

1120  ON V GOTO 1980,1880,2100,26
    70,2180,2290,2330,2380,2380,
    2650,1090,2400,2420,2180,267
    0,2710,2480,2530,2600,2690
1130  IF V <  > 10 THEN 1170
1140  IF N$ <  > "FLUTE" THEN 109
    0
1150  GOSUB 100: IF BE THEN  PRINT
    "THE BEAST SNORES EVEN MORE
    LOUDLY.": GOTO 1030
1160  PRINT "YOU HAVE SOOTHED THE
    SAVAGE BEAST! HE": PRINT "I
    S NOW SLEEPING PEACEFULLY.":
    BE = 1: GOTO 270
1170  IF BE = 0 AND (N$ = "SAFE" OR
    N$ = "LOCK" OR N$ = "BEAST" OR
    V = 11) AND (V <  = 2 OR V =
    4 OR V = 5 OR V = 11 OR V =
    14 OR V = 16) THEN  PRINT "T
    HE BEAST HAS EATEN YOU!": GOTO
    2630
1180  IF N$ <  > "BEAST" THEN 123
    0
1190  IF BE = 0 THEN 1090
1200  IF V = 5 THEN  PRINT "IT'S
    SLEEPING.": GOTO 1030
1210  IF V = 16 THEN 260
1220  GOTO 1090
1230  IF CH = 0 AND (N$ = "SAFE" OR
    N$ = "LOCK" OR V = 11) THEN
     PRINT "IT'S TOO HIGH TO REA
    CH!": GOTO 1030
1240  IF CH AND (V = 5 OR V = 4) AND
    (N$ = "SAFE" OR N$ = "LOCK")
     THEN  PRINT "IT'S A COMBINA
    TION LOCK! IT REQUIRES": PRINT
    "ONE TWO-DIGIT NUMBER FROM 0
    1-99.": GOTO 1030
1250  IF V <  > 11 THEN 1120
1260  IF  VAL (N$) <  = 0 THEN 10
    90
1270  IF  VAL (N$) <  > CO THEN  PRINT
    "WRONG COMBINATION!": GOTO 1
    030
1280  HCOLOR= 0: HPLOT 182,28 TO
    182,40: HCOLOR= 3: HPLOT 178
    ,28 TO 189,26 TO 189,42 TO 1
    78,40: HCOLOR= 1: FOR X = 16
    5 TO 177: HPLOT X,29 TO X,39
    : NEXT X
1290  PRINT CP$;CI$;"THE SAFE OPE
    NS AND INSIDE IS THE": PRINT
    "NICHE FORTUNE! YOU WIN!!";C
    N$;: VTAB 23: HTAB 1: FOR X =
    1 TO 5: PRINT  CHR$ (7);: NEXT
    X
1300  INPUT "DO YOU WANT TO PLAY
    AGAIN?(Y/N) ";Y$: IF Y$ = "Y
    " THEN  RUN
1310  GOTO 2620
1320  IF N$ <  > "ICEBOX" THEN 14
    20
1330  IF V <  > 4 THEN 1360
1340  IF IC THEN  PRINT "IT'S ALR
    EADY OPEN.": GOTO 1030
1350 IC = 1: GOTO 270
1360  IF V <  > 15 THEN 1390
1370  IF IC = 0 THEN  PRINT "IT'S
    ALREADY CLOSED.": GOTO 1030

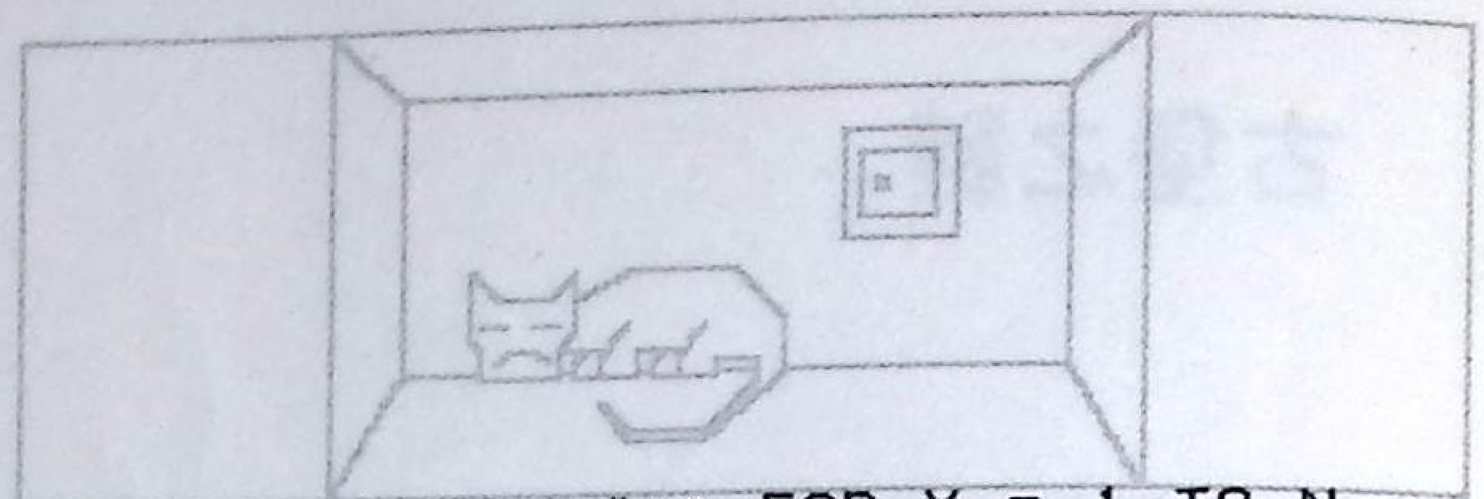
1380 IC = 0: GOTO 270
1390  IF V <  > 5 THEN 1440
1400  IF IC = 0 THEN  PRINT "IT'S
    CLOSED.": GOTO 1030
1410  PRINT "INSIDE IS A PASSAGE!
    ": GOTO 1030
1420  IF N$ <  > "PASSAGE" THEN 1
    510
1430  IF V = 5 THEN  PRINT "IT'S
    DARK.": GOTO 1030
1440  IF V = 1 THEN 1470
1450  IF N$ = "PASSAGE" THEN 1090

1460  GOTO 1120
1470  IF IC = 0 THEN  PRINT "IT'S
    CLOSED.": GOTO 1030
1480  IF RU = 0 THEN  PRINT "IT'S
    TOO COLD TO WALK IN THERE!"
    : GOTO 1030
1490 RM = 4: IF BE THEN  PRINT "A
    SAVAGE BEAST SLEEPS QUIETLY
    .": GOSUB 250: GOTO 270
1500  PRINT "A SAVAGE BEAST GUARD
    S THE SAFE!": GOSUB 250: GOTO
    270
1510  IF N$ <  > "STOVE" THEN 112
    0
1520  IF V <  > 5 THEN 1550
1530  IF ST THEN  PRINT "IT'S ON.
    ": GOTO 1030
1540  PRINT "IT'S OFF.": GOTO 103
    0
1550  IF V <  > 8 THEN 1590
1560  IF ST THEN  PRINT "IT'S ALR
```

古堡之謎



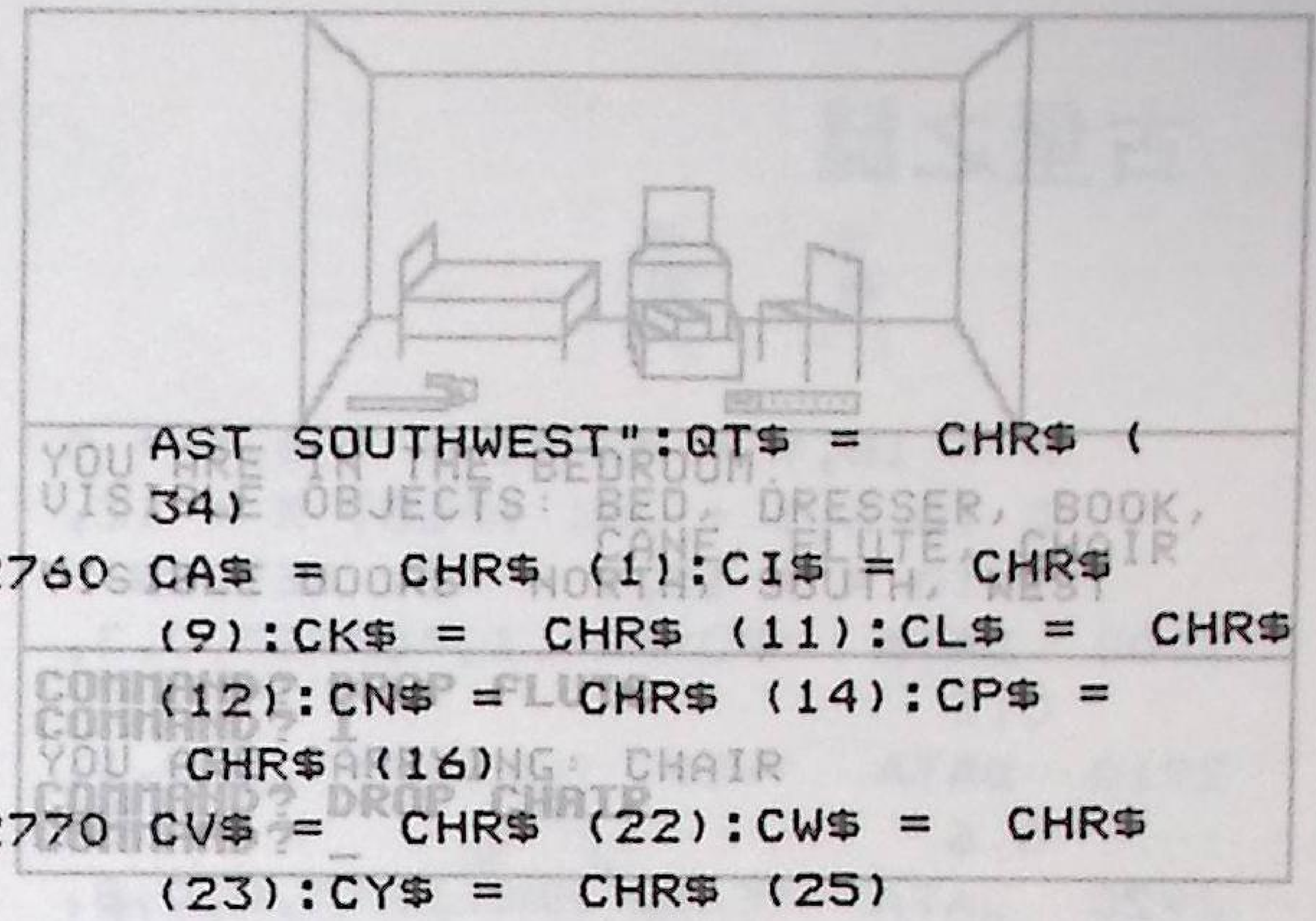

```
     EADY ON.": GOTO 1030
1570 ST = 1:N$ = "KETTLE": GOSUB
     170: IF F AND SO = 0 THEN SO
      = 1: PRINT "THE KETTLE IS N
     OW COOKING.": GOTO 1030
1580  GOTO 260
1590  IF V < > 9 THEN 1120
1600  IF ST = 0 THEN  PRINT "IT'S
      ALREADY OFF.": GOTO 1030
1610 ST = 0: GOTO 260
1620  IF N$ <  > "DRESSER" AND N$
      < > "DRAWER" THEN 1800
1630  IF V < > 4 THEN 1690
1640  IF DR THEN  PRINT "IT'S ALR
     EADY OPEN.": GOTO 1030
1650 N$ = "CANE": GOSUB 120: IF F
      = 0 THEN 1680
1660  PRINT "IT COMES OPEN! THE C
     ANE DID THE TRICK!";:DR = 1:
      IF DO = 0 THEN NO(RM) = NO(
     RM) + 1:O$(RM,NO(RM)) = "BOO
     K"
1670 GOTO 270
1680  PRINT "IT'S STUCK!": GOTO 1
     030
1690  IF V < > 5 THEN 1730
1700  IF DR = 0 THEN  PRINT "IT'S
      CLOSED.": GOTO 1030
1710 N$ = "BOOK": GOSUB 170: IF F
      AND BP = 0 THEN  PRINT "INS
     IDE IS A BOOK!": GOTO 1030
1720  PRINT "IT'S OPEN.": GOTO 10
     30
1730  IF V <  > 15 THEN 1120
1740  IF BP THEN DO = 1: GOTO 179
     0
1750  IF DR = 0 THEN  PRINT "IT'S
      ALREADY CLOSED.": GOTO 1030
1760 N$ = "BOOK": GOSUB 170: IF F
      = 0 THEN DO = 1: GOTO 1790
1770  IF F < NO(RM) THEN  FOR X =
      F TO NO(RM) - 1:O$(RM,X) = O
     $(RM,X + 1): NEXT X
1780 NO(RM) = NO(RM) - 1:DO = 0
1790 DR = 0: GOTO 270
1800  IF N$ < > "BED" AND N$ <  >
      "SHEET" THEN 1120
1810  IF V = 5 THEN  PRINT "THERE
     'S SOMETHING WRITTEN ON THE"
     : PRINT "SHEET!": GOTO 1030
1820  IF V = 13 THEN  PRINT "IT S
     AYS:": PRINT QT$;"LEFT-HAND
     DIGIT="; INT (CO / 10);QT$;"
     .": GOTO 1030
1830  IF N$ <  > "SHEET" THEN 187
     0
1840  IF V = 2 THEN  PRINT "IT'S
     TOO SECURELY FASTENED.": GOTO
     1030
1850  IF V = 16 THEN 260
1860  GOTO 1090
1870  GOTO 1120
1880  IF N$ <  > "SOUP" THEN 1930
1890 N$ = "KETTLE": GOSUB 170: IF
      F THEN 1910
1900  GOSUB 120: IF F = 0 THEN 19
     20
1910  PRINT "YOU CAN'T GET IT OUT
      OF THE KETTLE": PRINT "WITH
      YOUR HANDS.": GOTO 1030
1920  PRINT QT$;"SOUP";QT$;" ISN'
     T HERE.": GOTO 1030
1930  IF N$ = "DRAWER" THEN N$ =
     "DRESSER"
1940  GOSUB 150: IF NP = P8 THEN
      PRINT "YOUR HANDS ARE FULL.
     ": GOTO 1030
1950  IF N$ = "SAFE" OR N$ = "STO
     VE" OR N$ = "ICEBOX" OR N$ =
     "BED" OR N$ = "DRESSER" OR N
     $ = "TABLE" THEN  PRINT "IT'
     S TOO HEAVY TO LIFT.": GOTO
     1030
1960 NP = NP + 1:P$(NP) = N$: IF
     F < NO(RM) THEN  FOR X = F TO
     NO(RM) - 1:O$(RM,X) = O$(RM,
     X + 1): NEXT X
1970 NO(RM) = NO(RM) - 1: GOTO 27
     0
1980  IF N$ < > "CHAIR" THEN 201
     0
1990  IF CH THEN  PRINT "YOU'RE A
     LREADY ON THE CHAIR.": GOTO
     1030
2000 GOSUB 200:CH = 1: GOTO 260
2010  IF N$ <  > "FLOOR" AND N$ <
      > "DOWN" THEN 2040
2020  IF CH = 0 THEN  PRINT "YOU'
     RE ALREADY STANDING ON THE F
     LOOR.": GOTO 1030
2030 CH = 0: GOTO 260
2040  FOR D = 1 TO 4: GOSUB 240: IF
     N$ =  LEFT$ (DI$, LEN (N$)) THEN
     X = D:D = 4: NEXT : GOTO 206
     0
2050  NEXT D: PRINT "THERE'S NO S
     UCH DIRECTION.": GOTO 1030
2060  IF R(RM,X) = 0 THEN  PRINT
     "THERE'S NO ENTRANCE THERE."
     : GOTO 1030
```



```
2070  IF CH THEN  PRINT "YOU HAVE
      FALLEN FROM THE CHAIR AND":
      PRINT "BROKEN YOUR LEG!": GOTO
     2630
2080  IF RM = 4 AND RU = 0 THEN  PRINT
     "IT'S TOO COLD TO WALK IN TH
     ERE!": GOTO 1030
2090 RM = R(RM,X): GOTO 270
2100  IF N$ <  > "SOUP" THEN 2130
2110 N$ = "KETTLE": GOSUB 120: IF
     F THEN 1910
2120  PRINT "YOU DON'T HAVE ";QT$
     ;"SOUP";QT$;".": GOTO 1030
2130  GOSUB 100: IF F < NP THEN  FOR
     X = F TO NP - 1:P$(X) = P$(X
      + 1): NEXT X
2140 NO(RM) = NO(RM) + 1:O$(RM,NO
     (RM)) = N$:NP = NP - 1: IF N
     $ = "RUG" THEN RU = 0
2150  IF RM = 5 AND ST = 1 AND SO
      = 0 THEN SO = 1: PRINT "IT
     IS NOW COOKING ON THE STOVE.
     "
2160  IF RM = 8 AND N$ = "BOOK" THEN
     BP = 1
2170  GOTO 650
2180  IF N$ = "DOOR" OR N$ = "ENT
     RANCE" OR N$ = "WALL" THEN 2
     280
2190  IF N$ <  > "SOUP" THEN 2220
2200 N$ = "KETTLE": GOSUB 220: IF
     F = 0 THEN 1920
2210 N$ = "SOUP": GOTO 2250
2220  IF V = 14 THEN 1090
2230  IF N$ = "AROUND" OR N$ = N$
     (RM) OR N$ = "ROOM" THEN  PRINT
     "DO IT YOURSELF!": GOTO 1030
2240  GOSUB 200: IF N$ = "BOOK" THEN
      PRINT "THERE'S SOMETHING WR
     ITTEN IN IT!": GOTO 1030
2250  IF N$ = "SOUP" AND SO THEN
     2470
2260  IF N$ = "KETTLE" THEN  PRINT
     "THERE'S SOME SOUP IN IT!": GOTO
     1030
2270  IF N$ = "SOUP" THEN  PRINT
     "IT'S UNCOOKED.": GOTO 1030
2280  PRINT "IT'S AN ORDINARY ";N
     $;".": GOTO 1030
2290  IF N$ <  > "" THEN 1090
2300  PRINT "YOU ARE CARRYING";: IF
     NP = 0 THEN  PRINT " NOTHING
     .": GOTO 1030
2310  PRINT ": ";: FOR X = 1 TO N
     P: IF X > 1 THEN  PRINT ", "
     ;
2320  PRINT P$(X);: NEXT X: PRINT
     : GOTO 1030
2330  IF N$ <  > "KETTLE" AND N$ <
      > "SOUP" THEN 1090
2340 TN$ = N$:N$ = "KETTLE": GOSUB
     120: IF F = 0 THEN  PRINT "Y
     OU DON'T HAVE ";QT$;TN$;QT$;
     ".": GOTO 1030
2350  IF SO THEN  PRINT "IT'S ALR
     EADY COOKED.": GOTO 1030
2360  IF RM <  > 5 OR ST = 0 THEN
      PRINT "THERE'S NO HEAT HERE
     .": GOTO 1030
2370 SO = 1: GOTO 1030
2380  IF N$ = "STOVE" THEN  GOSUB
     150
2390  GOTO 1090
2400  IF N$ <  > "RUG" THEN 1090
2410  GOSUB 100:RU = 1: GOTO 260
2420  IF N$ <  > "BOOK" THEN 2440
2430  GOSUB 100: PRINT "IT SAYS:
     ";QT$;"MR. NICHE'S FAMOUS MO
     TTO": PRINT "WAS 'PUT YOUR M
     ONEY WHERE YOUR MOUTH": PRINT
     "IS.'";QT$: GOTO 1030
2440  IF N$ <  > "SOUP" THEN 1090
2450 N$ = "KETTLE": GOSUB 220: IF
     F = 0 THEN 1920
2460  IF SO = 0 THEN  PRINT "THE
     LETTERS ARE OBSCURED BY THE
     FROZENLIQUID.": GOTO 1030
2470  PRINT "IT'S ALPHABET SOUP!
     THE LETTERS SPELL": PRINT "O
     UT:": PRINT QT$;"RIGHT-HAND
     DIGIT=";CO - INT (CO / 10) *
     10;QT$;".": GOTO 1030
2480  IF N$ <  > "GAME" THEN 1090
2490  PRINT : PRINT D$;"OPENGAMEF
     ILE": PRINT D$;"DELETEGAMEFI
     LE": PRINT D$;"OPENGAMEFILE"
     : PRINT D$;"WRITEGAMEFILE"
2500  FOR X = 1 TO 9: PRINT NO(X)
     : IF NO(X) > 0 THEN  FOR Y =
     1 TO NO(X): PRINT O$(X,Y): NEXT
     Y
2510  NEXT X: PRINT NP: IF NP > 0
      THEN  FOR X = 1 TO NP: PRINT
     P$(X): NEXT X
2520  PRINT RM: PRINT CO: PRINT S
     T: PRINT SO: PRINT RU: PRINT
```

古堡之谜

```
    BE: PRINT IC: PRINT DR: PRINT
    CH: PRINT DO: PRINT BP: PRINT
    D$;"CLOSEGAMEFILE": GOTO 103
    0
2530  IF N$ < > "GAME" THEN 1090
2540  ONERR  GOTO 2590
2550  PRINT : PRINT D$;"OPENGAMEF
    ILE": PRINT D$;"READGAMEFILE
    "
2560  FOR X = 1 TO 9: INPUT NO(X)
    : IF NO(X) > 0 THEN  FOR Y =
    1 TO NO(X): INPUT O$(X,Y): NEXT
    Y
2570  NEXT X: INPUT NP: IF NP > 0
     THEN  FOR X = 1 TO NP: INPUT
    P$(X): NEXT X
2580  INPUT RM,CO,ST,SO,RU,BE,IC,
    DR,CH,DO,BP: PRINT D$;"CLOSE
    GAMEFILE": POKE 216,0: PRINT
    CP$;: GOSUB 250: GOTO 270
2590  PRINT D$;"DELETEGAMEFILE": POKE
    216,0: PRINT "THERE IS NO GA
    ME FILE ON THIS DISK.": GOTO
    1030
2600  IF N$ < > "GAME" THEN 1090
2610  INPUT "ARE YOU SURE?(Y/N) "
    ;Y$: IF Y$ < > "Y" THEN 103
    0
2620  PRINT CY$;CP$;: TEXT : PRINT
    : PRINT D$;"IN#0": PRINT D$;
    "PR#0": VTAB 1: PRINT "BYE!"
    : VTAB 12: PRINT D$;"FP"
2630  PRINT "BETTER LUCK NEXT TIM
    E...": PRINT : INPUT "DO YOU
     WANT TO TRY AGAIN?(Y/N) ";Y
    $: IF Y$ = "Y" THEN  RUN
2640  GOTO 2620
2650  IF N$ < > "FLUTE" THEN 109
    0
2660  GOSUB 100: GOTO 260
2670  IF N$ = "DRESSER" OR N$ = "
    DRAWER" OR N$ = "ICEBOX" THEN
     GOSUB 150
2680  GOTO 1090
2690  INPUT "ARE YOU SURE?(Y/N) "
    ;Y$: IF Y$ = "Y" THEN  RUN
2700  GOTO 1030
2710  IF N$ = "SOUP" THEN 1090
2720  GOSUB 200: IF F THEN 260
2730  GOTO 1090
2740 NV = 20:P8 = 2: DIM P$(P8),N
    $(9),NO(9),O$(9,8),R(9,4),NS
    (NV),VB$(NV,22)
2750 D$ =  CHR$ (4):DE$ = "NORTHE
```

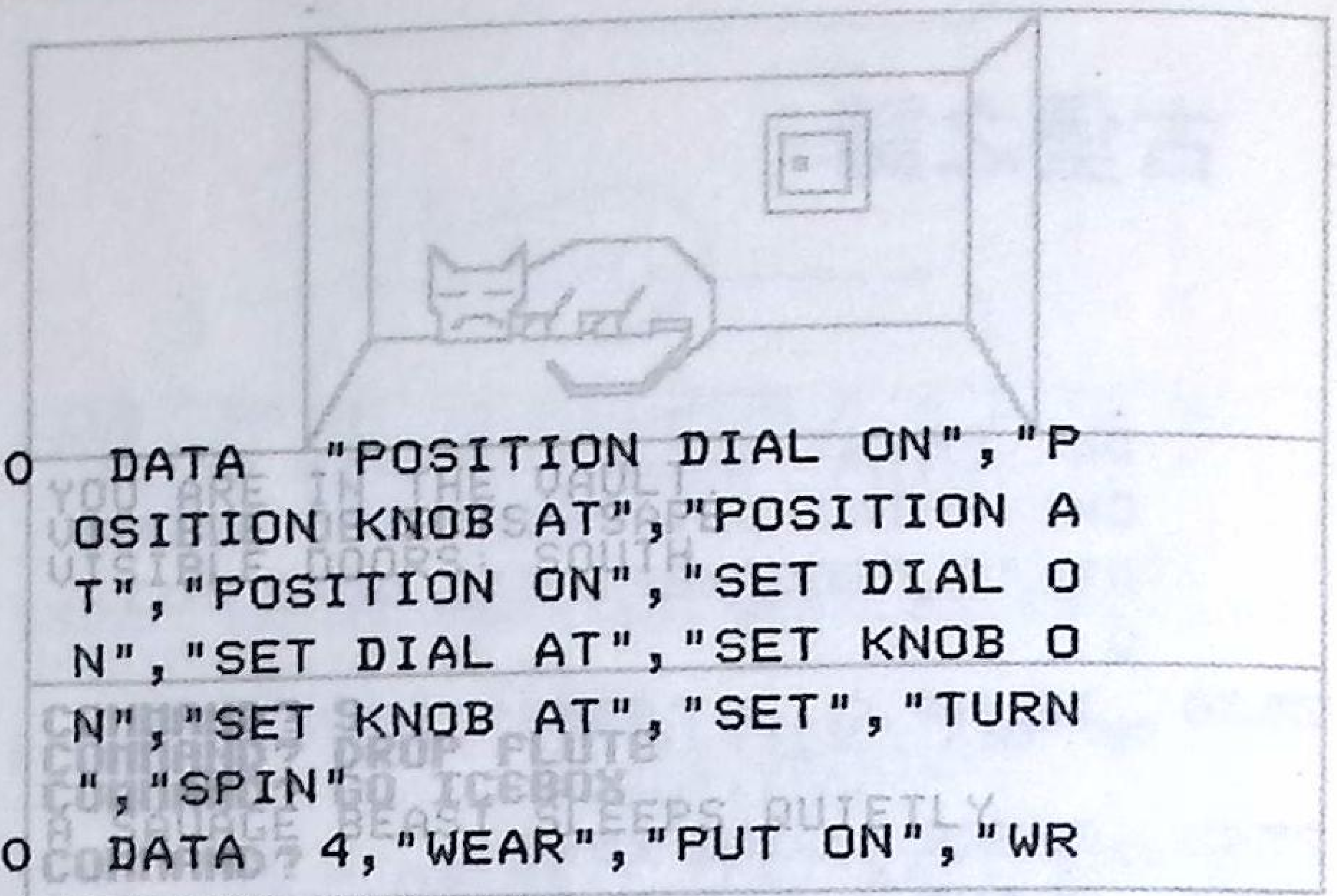


```
    AST SOUTHWEST":QT$ =  CHR$ (
    34)
2760 CA$ =  CHR$ (1):CI$ =  CHR$
    (9):CK$ =  CHR$ (11):CL$ =  CHR$
    (12):CN$ =  CHR$ (14):CP$ =
     CHR$ (16)
2770 CV$ =  CHR$ (22):CW$ =  CHR$
    (23):CY$ =  CHR$ (25)
2780  HCOLOR= 1: HPLOT 0,0: CALL
    62454: HCOLOR= 0: HPLOT 279,
    0 TO 279,191: HCOLOR= 3: HPLOT
    0,0 TO 278,0 TO 278,191 TO 0
    ,191 TO 0,0
2790 X = 16:Y = 16: GOSUB 2880:X =
    238:Y = 16: GOSUB 2880:X = 2
    38:Y = 127: GOSUB 2880:X = 1
    6:Y = 127: GOSUB 2880
2800  PRINT CY$;CI$;: VTAB 9: HTAB
    8: PRINT  SPC( 25): VTAB 10:
    HTAB 8: PRINT CA$;"2    C
    A S T L E    ": VTAB 11
    : HTAB 8: PRINT  SPC( 25)
2810  VTAB 13: HTAB 15: PRINT CN$
    ;CA$;"1";CL$;"***";CK$;" BY
    ***": VTAB 15: HTAB 4: PRINT
    CN$;CA$;"1";CL$;"********
    COMPUHOUSE        ********"
2820  FOR X = 1 TO 9: READ N$(X),
    NO(X): IF NO(X) > 0 THEN  FOR
    Y = 1 TO NO(X): READ O$(X,Y)
    : NEXT Y
2830  FOR Y = 1 TO 4: READ R(X,Y)
    : NEXT Y: NEXT X: FOR X = 1 TO
    NV: READ NS(X): FOR Y = 1 TO
    NS(X): READ VB$(X,Y): IF X <
    > 6 THEN VB$(X,Y) = VB$(X,Y
    ) + " "
2840  NEXT Y: NEXT X:RM = 1:X =  RND
    ( - PEEK (78)):CO =  INT (9
    9 * RND (1)) + 1
2850  VTAB 17: HTAB 9: PRINT "(H"
    ;CL$;"IT ANY KEY TO BEGIN)";
    CK$;CA$;"0";: WAIT  - 16384,
    128: POKE  - 16368,0
2860  PRINT CP$;: HPLOT 3,3 TO 27
    6,3 TO 276,187 TO 3,187 TO 3
    ,3: HPLOT 3,91 TO 276,91: HPLOT
    3,139 TO 276,139
2870  VTAB 19: HTAB 1: GOSUB 250:
     GOTO 270
2880  HPLOT X + 24,Y + 12 TO X +
    18,Y + 6 TO X + 6,Y + 6 TO X
    ,Y + 12 TO X,Y + 18 TO X + 6
    ,Y + 24 TO X + 18,Y + 24 TO
    X + 24,Y + 30
2890  HPLOT  TO X + 24,Y + 36 TO
```

古堡之謎

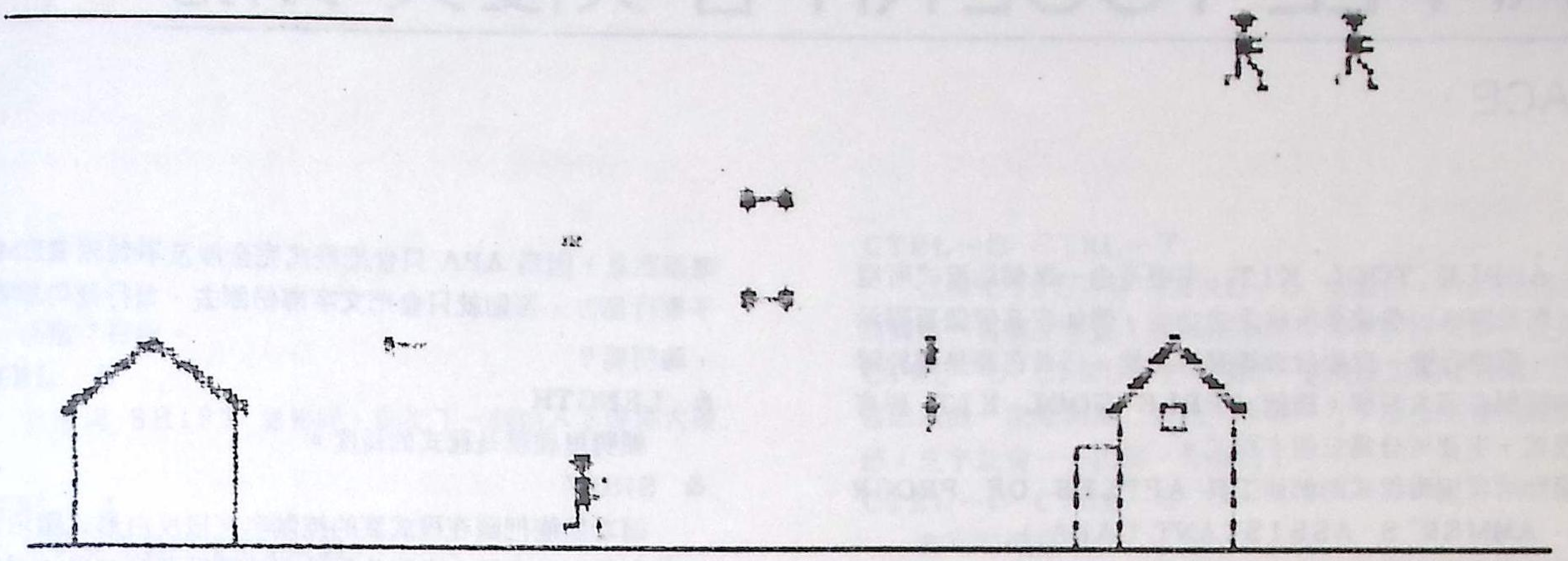
```
     X + 18,Y + 42 TO X + 6,Y + 4
     2 TO X,Y + 36: HPLOT X + 12,
     Y TO X + 12,Y + 48: RETURN
2900  DATA  "FOYER",1,"CANE",0,3,
     0,2
2910  DATA  "WEST HALLWAY",0,0,1,
     0,6
2920  DATA  "EAST HALLWAY",0,0,8,
     0,1
2930  DATA  "VAULT",1,"SAFE",0,0,
     5,0
2940  DATA  "KITCHEN",3,"KETTLE",
     "STOVE","ICEBOX",0,0,6,0
2950  DATA  "BALLROOM",1,"RUG",5,
     2,0,0
2960  DATA  "MUSIC ROOM",1,"FLUTE
     ",0,0,8,0
2970  DATA  "BEDROOM",2,"BED","DR
     ESSER",7,0,9,3
2980  DATA  "STUDY",2,"CHAIR","TA
     BLE",8,0,0,0
2990  DATA  16,"ENTER","GO TO","G
     O INSIDE","GO INTO","GO IN",
     "GO","CLIMB ONTO","CLIMB ON"
     ,"CLIMB INSIDE","CLIMB INTO"
     ,"CLIMB IN","CLIMB","STAND O
     N","STEP ONTO","STEP ON","ST
     EP"
3000  DATA  4,"GET","TAKE","PICK
     UP","GRAB"
3010  DATA  5,"DROP","PUT DOWN","
     SET DOWN","THROW DOWN","THRO
     W"
3020  DATA  5,"OPEN","PRY OPEN","
     PRY","PULL OPEN","PULL"
3030  DATA  6,"LOOK AT","LOOK INT
     O","LOOK IN","LOOK","EXAMINE
     ","SEARCH"
3040  DATA  2,"INVENTORY","I"
3050  DATA  3,"COOK","HEAT","BOIL
     "
3060  DATA  8,"TURN ON","SWITCH O
     N","ACTIVATE","LIGHT UP","LI
     GHT","FIRE UP","FIRE","IGNIT
     E"
3070  DATA  4,"TURN OFF","SWITCH
     OFF","SHUT OFF","DEACTIVATE"

3080  DATA  3,"PLAY","BLOW INTO",
     "BLOW"
3090  DATA  22,"DIAL","TURN TO","
     COMBINATION","SPIN COMBINATI
     ON","SPIN TO","SPIN DIAL TO"
     ,"SPIN KNOB TO","TURN DIAL T
     O","TURN KNOB TO","POSITION
     DIAL AT","POSITION KNOB ON"
```



```
3100  DATA  "POSITION DIAL ON","P
     OSITION KNOB AT","POSITION A
     T","POSITION ON","SET DIAL O
     N","SET DIAL AT","SET KNOB O
     N","SET KNOB AT","SET","TURN
     ","SPIN"
3110  DATA  4,"WEAR","PUT ON","WR
     AP ON","WRAP"
3120  DATA  1,"READ"
3130  DATA  3,"EAT","DRINK","CONS
     UME"
3140  DATA  2,"CLOSE","SHUT"
3150  DATA  3,"KILL","HIT","DESTR
     OY"
3160  DATA  2,"SAVE","STORE"
3170  DATA  2,"LOAD","RECALL"
3180  DATA  4,"QUIT","END","LEAVE
     ","STOP"
3190  DATA  1,"RESTART"□
10   REM **********************
20   REM *  ADVENTURE CASTLE  *
30   REM *   BY COMPUHOUSE    *
40   REM * COPYRIGHT (C) 1984 *
70   REM **********************
80   HIMEM: 8191
90   TEXT : HOME : HGR : POKE  - 1
     6302,0: ONERR  GOTO 140
100 ADRS = 0: PRINT  CHR$ (4);"BL
     OAD RBOOT": CALL 520:ADRS =
      USR (0),"HRCG": IF ADRS < 0
      THEN ADRS = ADRS + 65536
110 A = 2:CS = ADRS - 768 * A: HIMEM:
     CS:CH =  INT (CS / 256):CL =
     CS - 256 * CH: POKE ADRS + 7
     ,CL: POKE ADRS + 8,CH
120  FOR I = 1 TO A: READ CN$: PRINT
      CHR$ (4);"BLOAD";CN$;",A";C
     S + (I - 1) * 768: NEXT I
130  CALL ADRS + 3: POKE 216,0: GOTO
     160
140  TEXT : PRINT "UNABLE TO LOAD
     ": POKE 216,0: END
150  DATA FLOW.SET,GOTHIC.SET
160  ONERR  GOTO 190
170  PRINT  CHR$ (4)"EXEC CASTLE
     EXEC": END
180  REM  CREATE FILE IF NOT FOUN
     D
190  POKE 216,0:D$ =  CHR$ (4)
200  PRINT D$"OPEN CASTLE EXEC": PRINT
     D$"WRITE CASTLE EXEC"
210  PRINT "FP": PRINT "POKE 103,
     1": PRINT "POKE 104,64": PRINT
     "POKE 16384,0": PRINT "RUN C
     ASTLE"
220  PRINT D$"CLOSE": GOTO 170 □
```

高空擲物



這個遊戲名叫高空擲物，遊戲一開始便出現兩間屋，而自己則在其中一間屋，要由這裡行到對面的屋裡，看似簡單，但在這段路程中，會有很多東西跌下來的，而且那間屋是不會時常開門的，且也不需要驚慌，只要你能避過高空擲下來的物體，便可得到獎分。希望你們能順利到達目的地。

本遊戲玩法十分容易，只需按→鍵前進和←鍵退後。在程式磁碟中的檔案名字是 DANGER ROAD。□

```
1   REM * * * * * * * * * *
2   REM *   DANGER ROAD   *
3   REM * BY:   CHI-MAN   *
4   REM * * * * * * * * * *
5   REM
10  HIMEM: 36860: HOME : VTAB 23:
       PRINT "SCORE :";
20 D$ = "100000":R = 36864:W = -
       16336:D = 1
30  HGR : SCALE= 1: HCOLOR= 3: ROT=
    0: GOSUB 3000
220  READ A: IF A = 999 THEN  POKE
    232,0: POKE 233,144: GOTO 30
    0
230  POKE R,A:R = R + 1: GOTO 220
300  GOSUB 940
600 RD =  INT ( RND (1) * 4) + 1:
    A$(RD) = A$(RD) + "7": XDRAW
    RD + 2 AT 32 + RD * 33,106 -
    7 * 9
680  FOR T = 1 TO 4:LA =  LEN (A$
    (T)): IF LA = 0 THEN 750
690 B$ = A$(T)
700  FOR C = 1 TO LA:L$ =  MID$ (
    A$(T),C,1):L =  VAL (L$)
710  IF L = 2 AND T = J THEN P =
    T:Q = L: GOSUB 1090: GOTO 73
    0
720  XDRAW T + 2 AT 32 + T * 33,1
    06 - L * 9
723 A =  VAL (B$) -  VAL ( LEFT$
    (D$,LA - C + 1)): IF A = 0 THEN
    B$ = "":MA = MA + .3: GOTO 7
    45
724 B$ =  STR$ (A)
725  IF L = 1 THEN MA = MA + .3: GOTO
    730
726 L = L - 1
727  XDRAW T + 2 AT 32 + T * 33,1
    06 - L * 9:DE = 2: GOSUB 200
    0
730  GOSUB 1200: HTAB 10:KK =  INT
    (MA + .1): PRINT KK;
740  NEXT
745 A$(T) = B$
750  NEXT T
760  IF GE < MA - 3 AND J = 0 THEN
     GOSUB 840
800 AB =  LEN (A$(1) + A$(2) + A$
    (3) + A$(4)): IF AB > 12 THEN
    NS = 0
810  IF AB < NS / 3 + 1 THEN 600
820  GOTO 680
840  GOSUB 940:J = J + 1 -  INT (
    J / 5): IF J = 5 THEN 1500
850  GOTO 940
900  GOSUB 940:J = J - 1 +  INT (
    (6 - J) / 5)
940  XDRAW 1 AT 32 + J * 33,99: RETURN
1090 CT = CT + 1:DE = 20: GOSUB 2
    000: XDRAW 1 AT 200 + 20 * C
    T,20: IF CT = 3 THEN 4060
1100 DE = 5: GOSUB 2000: XDRAW P +
    2 AT 32 + P * 33,106 - Q * 9
    : XDRAW 1 AT 32 + J * 33,99
1110  XDRAW 2 AT 28 + P * 33,106:
    DE = 10: GOSUB 2000: XDRAW 2
    AT 28 + P * 33,106:J = 0:GE
    = MA: XDRAW 1 AT 32 + J * 3
    3,99: IF  LEN (B$) = 1 THEN
    B$ = "": GOTO 1130
1120 B$ =  RIGHT$ (B$, LEN (B$) -
    1)
1130  RETURN
```

（下轉第41頁）

APPLE TOOL KIT 各項優異功能

ACE

APPLE TOOL KIT 主要是由一組輔助程式所組成，幫助解決一些蘋果所缺乏之功能，例如在高解像頁顯示文字，重整行號，自動行數等實用功能，以前在蘋果難於解決的問題亦迎刃而解。總括 APPLE TOOL KIT 所有的程式，主要可分爲三個主程式。

①協助用家編輯程式的輔助工具 APPLES OF PROGR—AMMER'S ASSISTANT（APA）。

②高解文字產生器，HI—RES CHARACTER GENE—RATOR（HRCG）和一個用於高解文字產生器的字形編輯程式 ANIMATRIX。

③匯編語言編寫程式 APPLE 6502 ASSEMBLER／EDITOR。

由於專供編寫匯編語言的程式亦不少，所以暫且不談匯編語言編寫器這程式，相對地 APA 及 HRCG 所提供的功能，就顯得非常有效和強大，對於各位熱衷於編寫程式的同志們就不容錯過。

APA 所提供之功能

& AUTO <START>〔,<INC>〕

爲自動行數功能，用家只需輸入開始行號，每次按 R—ETURN便會自動把行數加10，並把行號顯示，但如果要以不同的數字遞增，只需鍵入指令，開始行號以及遞增數字便可。

& MANUAL

將自動行數輸入功能取消。

& RENUMBER〔<START>〕〔,〔<INC>〕〔,〔<FIRST>〕〔,<LAST>〕〕〕

如果只是想用修改完的程式使其行號整齊些，只需鍵入&RENUMBER 便可，但如要改動某部份程式或有特別的要求，則要把其他參數一起用才能。其中 FIRST 及 LA—ST 是些指定需要重整行號的那部份程式的起始行號及結尾行號，而 START 及 INC 則是重整之後起始行號經改變爲的行號和遞增數值。

& HOLD

這功能就是把正在記憶裡的程式搬到 HIMEM 上隱藏起來，這時用家可以載入另外的一個程式於記憶中，以便和隱藏着的程式合二爲一。

& MERGE

當用家把記憶中的程式編輯妥當，就可以用這功能把剛才用 & HOLD 藏起的程式合二爲一，但要注意若有相同行號出現時，APA 會詢問是否繼續，如繼續的話，藏起的那個程式會被首選，而正在記憶的那一行就會被刪去。

& COMPRESS

所謂把程式壓縮就是把 REM 句子刪去，以減少記憶空間及使程式運行得迅速些，但不要担心壓縮後的程式會出現錯誤訊息，因爲 APA 只會把程式完全涉及不到的 REM 句子整行刪去，否則就只會把文字部份刪去，而行號仍然存在，聰明嗎？

& LENGTH

顧明思義就是程式的長度。

& SHOW

這功能能把藏在程式裏的控制字符以反白形式顯示出來，這對於利用高解文字產生器的程式顯得非常有用，因爲這些程式通常要用到很多控制字符，但對於把這類程式輸出印字機亦是無往而不利，因爲在這狀態下的控制是會失去其功用。

& NOSHOW

回復正常狀態，令所有控制字符再次有效。

& KEYS

讓用家多按三個字符，CTRL—O，"—"，CTRL—L，"＼'，CTRL—K，"〔"

& XREF

就是把所有變數連同它們在那一行出現一併列出，這對打錯變數名稱的一種蟲，很快就可以查出，又如果閣下想分析一下別人的程式，這功能也是用得着的。

HRCG 所提供之功能

CTRL—An

選擇不同的字形組，其中n爲字形組的參數值，1至9爲用家自選字形，0爲標準字形，但使用自選字形時，必須把字形資料先存入記憶，稍後會詳細提及。

CTRL—N

正常顯示模式，印黑底白字。

CTRL—I

反白顯示模式。

CTRL—Q

將游標移到螢幕的左上角，但不清除畫面。

CTRL—P

清除畫面。

CTRL—E

將整行清除。

CTRL—F

清除畫面，並將游標移到螢幕的右下角。

CTRL—Y

設定整個文字窗。

CTRL—V

設定左上文字窗。

CTRL—W

設定右下文字窗。

CTRL—K

APPLE TOOL KIT 各項優異功能

大楷字符鎖。

CTRL—L

小楷字符鎖。

CTRL—S

效果與 SHIFT 鍵相同，即把下一個輸入字符爲大楷。

CTRL—Z

把所有參數回復起始狀態。

CTRL—B，CTRL—D，CTRL—C

這三個指令，通常是相輔相成地使用，就好像我們要畫一個人形，我們就可以用方塊模式輸入，首先用 ANIMA—TRIX 把左手存入A字符，右手存入B字符，左脚存入C字符，右脚存入D字符，接着

```
        CTRL—B      CTRL—C
           ↓         ↙
10 A$ = "   AB
            CD  "
             ↑
          CTRL—D

20 PRINTA$
```

這樣人形就會被印出，CTRL—B 開始方塊模式，CTRL—C 將游標移到下一行，並置於開始方塊模式的那個字符之下，CTRL—D 結束方塊模式。

CTRL—O CTRL—A

選擇高解第一頁爲文字頁

CTRL—O CTRL—B

選擇高解第二頁爲文字頁，但要注意選擇文字頁並沒有把畫面顯示轉到那頁。

CTRL—O CTRL—D

則就是把畫面顯示轉到文字頁那裡。

CTRL—O CTRL—C

即與文字頁內的背景，以反白形式顯示，亦即是把文字和背景作一次 EOR 計算並顯示。

CTRL—O CTRL—O

即與文字頁內的背景，重叠後並顯示，亦即是把文字和背景作一次 AND 計算然後顯示。

CTRL—O CTRL—P

回復正常文字顯示，這時文字頁的背景是會被文字所破壞了。

CTRL—O CTRL—R

這與 CTRL—O CTRL—C 的動作大致相同，但背景資料則由另外的一頁取得。

CTRL—O CTRL—T

這與 CTRL—O CTRL—O 相類似，例如我們選擇高解第一頁爲文字頁，在高解第二頁則有一行文字，使用 CTRL—O CTRL—T，再把一隻飛機從左邊飛到右邊，當然飛機一定要和第二頁的文字原一行，這樣隨着飛機的尾部，文字就會一一出現，有趣嗎？

CTRL—O CTRL—S

許可回捲顯示。

CTRL—O CTRL—W

禁止回捲顯示，作用當然是保護畫面內圖文

CTRL—O CTRL—Y，CTRL—O CTRL—Z，分別爲兩個用家自設副程式，而副程式的起始點就分別要放在 HRCG 起始點之後的11—14個地址。如 HRCG 的起始爲 $6000 ，則副程式A起始點就要存入 $600A，$600B，副程式B的起始點就要存入 $600D，$600E。

如何載入 APA 及 HRCG？

在 APPLE TOOL KIT 內分別有兩個程式稱爲 LOADAPA 及 LOADHRCG ，但使用起來總有點拖泥帶水的感覺，有時又會出現甚麽 UNABLE TO LOAD 等錯誤訊息，使人不明所以，而每次載入都要花二十秒左右時間，但主程式只有2，3K這麽短，爲何會花這麽久的時間呢？其實由於早期推出的 APPLE 電腦是有幾個不同記憶容量的版本（16K，32K，48K），所以爲着適合不同版本的用家，APA 和 HRCG 都可隨意搬到任何地址裡工作，而負責載入的程式就是 RBOOT 和 RLOAD ，就因爲每次都要經過計算才能搬到適當的位置，所以顯得非常緩慢，要解決這問題並不困難，因爲每次經由 RBOOT，RLOAD 載入的主程式，都是緊貯在 HIMEM 之下，所以只要把主程式轉爲普通的MC 程式便行。如我們喜歡把 HRCG 放在$6000 之下：

①鍵入 HIMEM：24576（$6000）

②RUN LOADHRCG

③BSAVE HRCG・OBJ，A$57FF，L$801

以後要用高解產生器，就只須載入 BLOAD HRCG・OBJ ，然後 CALL 22527（$57FF），同樣 APA 亦可以這樣做，而它們的程式長度分別爲 HRCG，$801 APA，$E1F

用家必須緊記兩點：

①不要 BRUN 這些MC 程式，只可用 BLOAD ，再 CALL 其起始點才行。

②若不預先定下 HIMEM，主程式是會緊貼在磁碟操作系統之下。

字形編輯功能

APPLE TOOL KIT 各項優異功能

在 APPLE TOOL KIT 內已包含多種字體，美術字體，圖形等就非使用 ANIM ATRIX 不可，自己動手做到自己滿意爲止，那就比 VIC，ATARI 等標準圖案來得靈活，雖然這程式還算易用，但是 JOYSTICK 輸入部份就令人非常反感，總是不能準確地把游標移到想到的地方，其實修改一下效果就截然不同了。

```
490 PL =  - 1
500 PX = PX +  SGN ( INT ( PDL (0
    ) / 85 - 1)): IF  ABS (PX -
    24) > 24 THEN PX = PX -  SGN
    (PX - 24)
510 PY = PY +  SGN ( INT ( PDL (1
    ) / 85 - 1)): IF  ABS (PY -
    19.5) > 19.5 THEN PY = PY -
     SGN (PY - 19.5)
```

但要眞正使用 HRCG 的用家自選字形除了把 HRCG 載入記憶，還要加點功夫，以下就是一個輔助計算地址的程式。

```
10  INPUT " HRCG STARTINT ADRS";A
    DRS
20  INPUT "  HOW MANY CHR,SET";A
30  IF A < 0 OR A > 9 THEN 20
40  IF ADRS < 0 THEN ADRS = ADRS +
    65536
50 CS = ADRS - 768 * A
60 CH =  INT (CS / 256) = CL = CS
     - 256 * CH
70  PRINT "HIMEN:"CS
80  PRINT "POKE"ADRS + 7","CL
90  PRINT "POKE"ADRS + 8","CH
100  FOR I = 1 TO A: PRINT "SET<"
    I">"CS + (I - 1) * 768: NEXT
```

設定 HIMEM，再 POKE 入那兩個數，然後再載入字形資料，用家自設字形就能以 CTRL—An 來選擇。相信各位都不會否認 APPLE TOOL KIT 的充滿着潛能，只要各位能夠花多點心思，妙用還是無窮無盡的，努力吧。□

有趣的排序

這裡提供一個排序程式，大家可視之爲一個非常有趣的遊戲程式。

程式會設在一個有一百個元素的列陣，卽 A（1）：1，A（2）＝2，……A（100）＝（100）。然後程式會將這個列陣在高解像頁裡繪劃出來（繪劃的結果是一條對角綫）。

程式最有趣的地方這時開始出現。程式是利用泡沫排序法（BUBBLE SORT）將各元素排序，當元素在浮動時，螢幕會顯示出它的運動情況。雖然泡沫排序法並非最快的速度排序方法，但在視覺娛樂上，它却是最有趣的一個。當然，可以不用這個排序法而隨意加插自己的排序子程式在程式尾。□

FUNNY SORTING

```
70  HGR : HCOLOR= 3
80  DIM A(100): HOME : VTAB 22: PRINT
    "CREATING ARRAY": FOR I = 1 TO
    100:A(I) = I: NEXT I
90  HGR : FOR I = 1 TO 100: HPLOT
    I,A(I): NEXT I
100  HOME : VTAB 22: PRINT "SHUFF
    LING": FOR I = 1 TO 100:B =
     INT ( RND (1) * 100) + 1:T =
    A(B):A(B) = A(I):A(I) = T: NEXT
    I
110  HGR : FOR I = 1 TO 100: HPLOT
    I,A(I): NEXT I
120 N = 100
130  HOME : VTAB 22: PRINT "SORTI
    NG": PRINT ""
140  GOSUB 200
150  PRINT "": HOME : VTAB 22: PRINT
    "DONE": END
200 N = N - 0
210 K = N - 1
220  FOR I = 1 TO K
230 L = N - I
240  FOR J = 1 TO L
250  IF A(J) < A(J + 1) THEN 290
260 T = A(J):A(J) = A(J + 1):A(J +
    1) = T
270  HCOLOR= 0: HPLOT J,0 TO J,10
    0: HCOLOR= 3: HPLOT J + 1,A(
    J + 1)
290  NEXT : NEXT
300  RETURN
```
□

混合機械語言/APPLESOFT程式的技術探討

第８期之高解像頁處理工具箱提供了一個6502的新技術，它將 APPLESOFT 與機械語言同時混合成爲一個新的程式，機械語言形式出現，使6502將它視作爲機械語言程式，我們只要加入一些技巧便可用作保護程式之用，當然包括更改 LIST，RESET 等命令的指向器內容，這方面筆者留給有興趣的讀者自己研究。

現在讓我們討論如何將機械語言及 APPLESOFT 混和成一新的機械語言程式。

當我們預備將兩種語言的程式合併時，首先及最重要的一步是在機械語言的網路上加入以下的網路：

```
JSR$FC58（清理螢幕）
LDA#$??  （APPLESOFT 程式的
             高位元開始位址）。
STA$68
LDA#$??  （APPLESOFT 程式的
             低位元開始位址）。
STA$67
LDA$#$!!（APPLESOFT 程式的
             低位元結束位址）。
STA$AF
STA$69
STA$6B
STA$60
LDA$!!   （APPLESOFT程式的
             高位元結束位址）。
STA$B0
STA$6A
STA$6C
STA$6E
RTS
```

這些命令能促使 APPLESOFT 知道新的程式開始及結束位址，使在執行時正確無誤，包括使變數及陣列存放在跟隨 APPLESOFT 程式結束的記憶空間裏，不過此種現像，一旦在新程式載入或直接執行時，使能令指向器回復原來的狀態， BLOAD 一個機械語言程式却不能有效地改變指向器的位置，因此必須依靠人手來完成，詳情可參考APPLESOFT BASIC PROGRAMMING REFEREN—CE MANUAL 的第140頁。

以上所說的指向器內存數值是結束位址，分別貯於$AF及 $B0之內，因此可以進入監督程式內檢查之。

另外，APPLESOFT 要求程式的首個 BYTE 的數值爲 $00 ，否則便無法執行。在加上機械語言在同一程式上時需在啓動網路（35 十進制或 $23 十六進制）的數值加上1來，取代 $00 BYTE 的數值，目的是使 APPLE—SOFT 知道需要保留多少空間給予機械碼貯存。

完成以上的步驟後，（卽加入上述的網路在機械語言中）你需決定保留的空間長度，方法是在 APPLESOFT 程式的起始部份加上多個 REM 敍述句，一句敍述句會佔據5個 BYTE，不過需在每行裏只存入一個 REM 敍述句。

這種技術的應用方法亦不太難，簡單來說是先鍵入你的 APPLESOFT 程式，緊記在開始時先鍵入一個長的 RE—M 敍述句，例如在第八期中的〔高解頁工具箱〕，我們在 APPLESOFT 程式前加入0 REM /////……（共189個）其次加上一段改 APPLESOFT 爲機械語言的子程式在內，如列表A所示，之後便鍵入上述的啓動程式。（此程式乃一機械語言之網路），它將覆蓋了你的 APPLE—SOFT 程式第0行，按800G使會令 APPLESOFT忘記其存在，完成後將這一切以 BSAVE 命令貯存起來。

經過這個方法處理的程式，若將程式指向器的數值更改仍然可以正常執行。因爲它會被當作爲機械語言程式貯存在記憶體內。故此 NEW，LOMEM，HIMEM，LOAD及 RESET 仍不會使本程式被破壞。當然 BOOT 機及 FP 並不包括在內。□

適合4至8歲小孩玩的遊戲

捉迷藏

遊戲有兩個程式列表：(1) LION HUNTING及(2) LION。程式磁碟上的FILENAME 亦一樣。

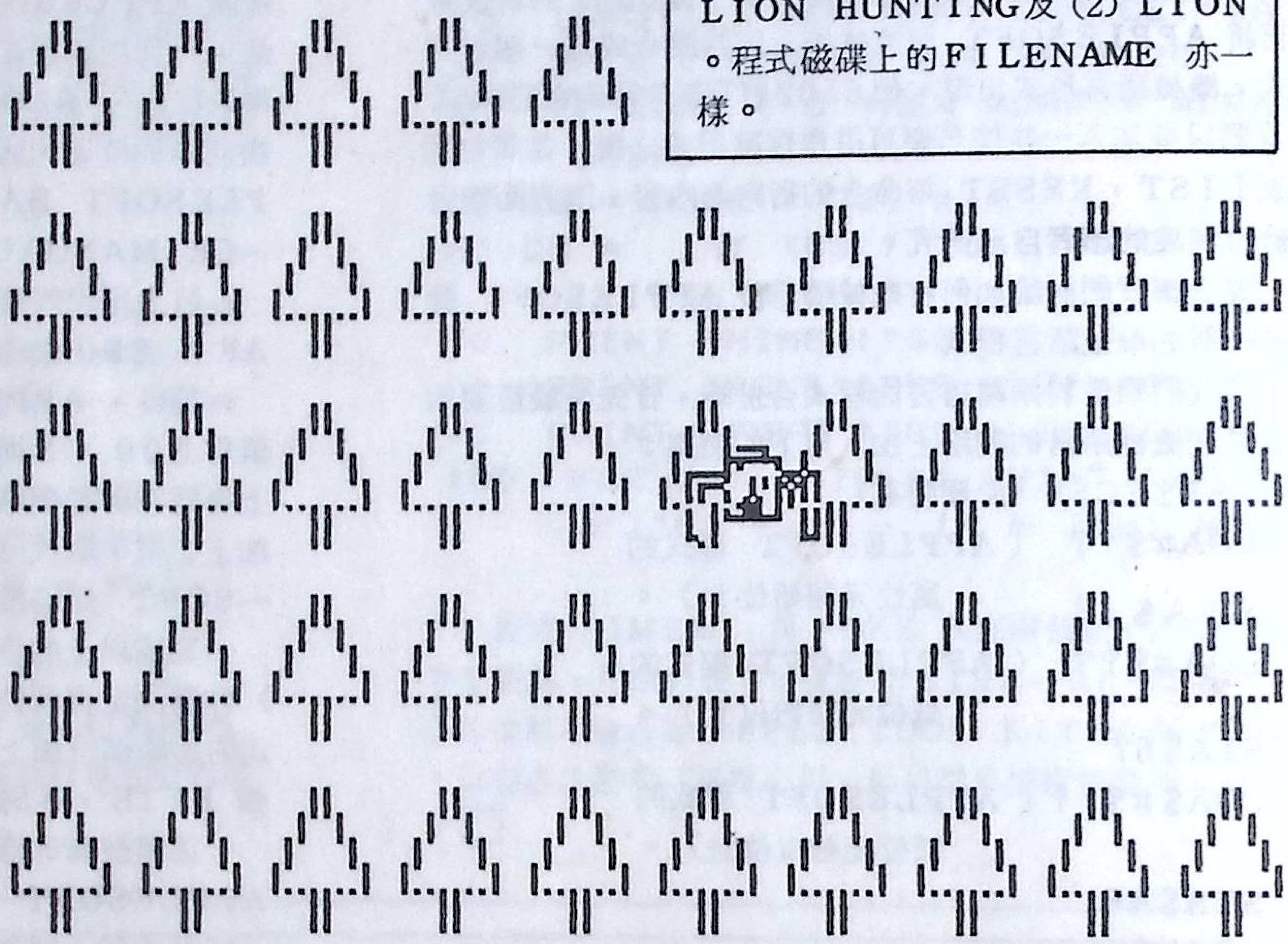
本遊戲適合4至8歲的兒童玩，他們會發覺遊戲非常有趣。遊戲要求兒童捉拿匿藏在森林內的天外怪獸。

這隻天外怪獸會跟孩子們開玩笑，他會在森林四週找尋可匿藏的地方，然後和孩子們玩起捉迷藏。

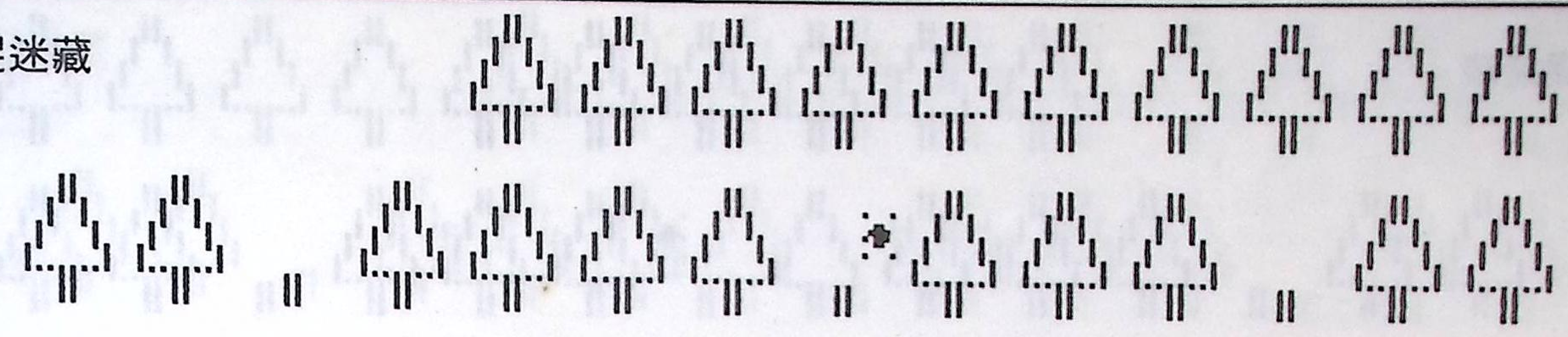
孩子可用U・D・L・R來移動（由左上角森林開始，每次移動步數不限）。

每次移動後，所處位置的那棵樹將會消失，同時電腦會發出五個音符的樂聲，假如孩童所處的位置越接近天外怪獸，則聲調越高。每回合遊戲，最多只能移動一次，否則會被怪獸吃去。

遊戲有幾種難度選擇，適合不同年齡的孩童。在高難度的版本中，天外怪獸匿藏的位置會隨時間而改變。（在此情況，最多可移動十四步）。



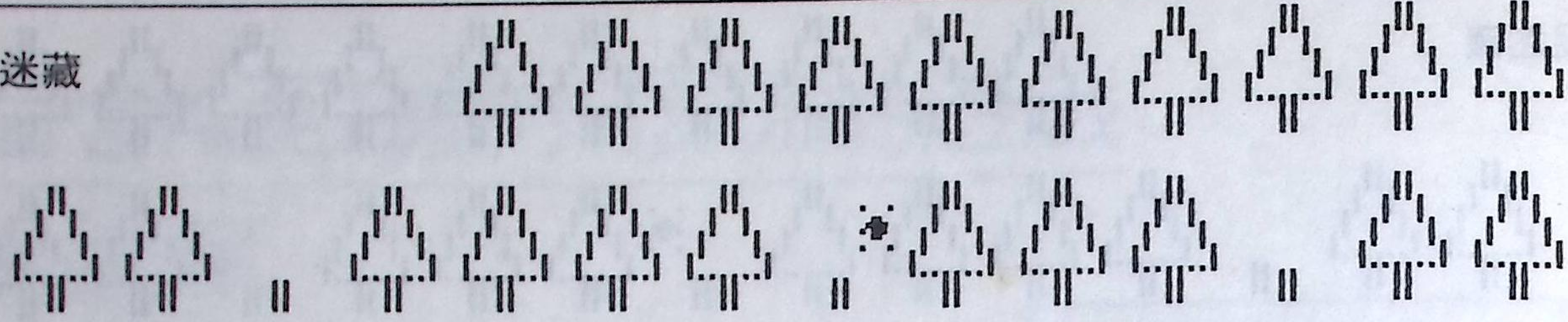
```
1 REM ****************************
2 REM *                          *
3 REM *        LION HUNTER       *
5 REM *                          *
6 REM ****************************
10 LOMEM: 23000: HOME : VTAB 10:
    HTAB 14: INVERSE : PRINT "L
   ION HUNTER";: NORMAL : FOR I
    = 1 TO 1000: NEXT I
20 GOSUB 1650:D$ = CHR$ (4): PRINT
    D$"BLOAD LION,A$5600"
30 POKE 232,0: POKE 233,86
40 GOSUB 1900
50 L = 0:PP = 0:W$ = "":M =  INT
    ( RND (1) * 10) + 4
60 F = 1: HOME : TEXT :SS = 10
70 HOME
80 VTAB 10
90 PRINT "A SPACE LION IS LANDIN
   G IN THE FOREST "
100 PRINT "IF YOU CAN GUESS WHER
   E HE IS HIDING ": PRINT "WIT
   HIN ";SS;" GUESSES, YOU WIN!
   !"
110 PRINT : INVERSE
120 PRINT "IF NOT, HE WILL EAT Y
   OU!!!"
130 NORMAL : PRINT
140 PRINT : INPUT "WHAT IS YOUR
   NAME?";N$
150 IF  LEFT$ (N$,2) =  LEFT$ (J
   $,2) THEN PP = 1
160 PRINT
170 PRINT : PRINT "HERE COMES TH
   E LION NOW...."
180 FOR N = 0 TO 1000: NEXT N
190 HGR : HCOLOR= 3: SCALE= F: ROT=
   0
200 NORMAL
210 FOR N = 0 TO 150
220 X =  RND (1) * 270
230 Y =  RND (1) * 150
240 HPLOT X,Y: HPLOT X + 1,Y
250 NEXT N
260 X =  INT ( RND (1) * 260)
270 FOR Y = 10 TO 140 STEP 2
280 XDRAW 1 AT X,Y
290 GOSUB 1690
300 XDRAW 1 AT X,Y
310 NEXT Y
320 SCALE= F
330 CALL 62450: SCALE= 2: HCOLOR=
   1
340 FOR X = 11 TO 271 STEP 20
350 FOR Y = 10 TO 150 STEP 30
360 DRAW 3 AT X,Y
370 DRAW 4 AT X,Y
380 NEXT Y
390 NEXT X
400 SCALE= 1: HCOLOR= 3
```



```
410  FOR A = 1 TO 25
420 X =  INT ( RND (1) * 271)
430 Y =  INT ( RND (1) * 150)
440  XDRAW 1 AT X,Y
450  GOSUB 1710
460  FOR N = 1 TO 300: NEXT N
470  XDRAW 1 AT X,Y
480  NEXT A
490  XDRAW 1 AT 140,150
500  VTAB 21
510  PRINT "I BET YOU CAN'T FIND
     ME....."
520  GOSUB 1800: GOSUB 1800: GOSUB
     1800
530  FOR N = 1 TO 3000: NEXT N
540  XDRAW 1 AT 140,150
550  HCOLOR= 3: XDRAW 2 AT 11,7
560  HOME : VTAB 21
570  PRINT "YOU ARE THE WHITE SQU
     ARE AT THE TOP LEFT"
580  VTAB 22: PRINT "TREE...THE C
     LOSER YOU GET TO THE LION "
590  VTAB 23: PRINT "THE HIGHER T
     HE SOUND..."
600  INPUT "HIT RETURN TO CONTINU
     E";Q$: HOME
610 XX = 11:YY = 10
620 X = 11 +  INT ( RND (1) * 14)
     * 20
630 Y = 10 +  INT ( RND (1) * 5) *
     30
640  SCALE= 1: IF L = SS THEN 152
     0
650  VTAB 22
660  PRINT "DO YOU WANT TO GO: L
     LEFT     U UP"
670  HTAB 20
680  PRINT "R RIGHT   D DOWN "
690  IF SS - L = 1 THEN   VTAB 24:
     INVERSE : PRINT " ";: PRINT
     SS - L;" MOVE REMAINING";: NORMAL
     : VTAB 23: HTAB 38: GOTO 710

700  VTAB 24: INVERSE : PRINT " "
     ;: PRINT SS - L;" MOVES REMA
     INING";: NORMAL : VTAB 23: HTAB
     38
710  GET D$: HOME : VTAB 21: IF D
     $ = "U" OR D$ = "D" OR D$ =
     "L" OR D$ = "R" THEN 730
720  PRINT  CHR$ (7): PRINT  CHR$
     (7): PRINT  CHR$ (7): GOTO 6
     50
730  INPUT "HOW MANY TREES DO YOU
      WANT TO MOVES?";Q$:Q =  VAL
     (Q$): IF  NOT Q THEN  HOME :
      VTAB 21: GOTO 730
740  IF PP = 1 THEN  XDRAW 1 AT X
     ,Y: FOR N = 1 TO 15: NEXT N
750  IF PP = 1 THEN  XDRAW 1 AT X
     ,Y
760  XDRAW 2 AT XX,YY - 3
770 L = L + 1
780  IF L = M AND  NOT PP THEN  GOSUB
     1450: XDRAW 2 AT XX,YY - 3: HOME
     : GOTO 620
790  IF D$ = "L" THEN  GOTO 830
800  IF D$ = "R" THEN  GOTO 920
810  IF D$ = "U" THEN  GOTO 1010
820  IF D$ = "D" THEN  GOTO 1110
830  HOME : VTAB 21
840  IF XX - 20 * Q < 11 THEN  PRINT
     "YOU CAN'T LEAVE THE FOREST,
      TRY AGAIN": XDRAW 2 AT XX,Y
     Y - 3: GOTO 650
850  FOR N = XX TO XX - 20 * Q STEP
      - 1
860  XDRAW 2 AT N,YY - 3: FOR A =
     1 TO 25: NEXT A: XDRAW 2 AT
     N,YY - 3: NEXT N
870 XX = XX - 20 * Q
880  HCOLOR= 0: SCALE= 2: DRAW 3 AT
     XX,YY: SCALE= 1
890  HCOLOR= 3: XDRAW 2 AT XX,YY -
     3
900  GOSUB 1220
910  GOTO 640
920  HOME : VTAB 21
930  IF XX + 20 * Q > 280 THEN  PRINT
     "YOU JUST LEFT THE FOREST,TR
     Y AGAIN": XDRAW 2 AT XX,YY -
     3: GOTO 650
940  FOR N = XX TO XX + 20 * Q
950  XDRAW 2 AT N,YY - 3: FOR A =
     1 TO 25: NEXT A: XDRAW 2 AT
     N,YY - 3: NEXT N
960 XX = XX + 20 * Q
970  HCOLOR= 0: SCALE= 2: DRAW 3 AT
     XX,YY: SCALE= 1
980  HCOLOR= 3: XDRAW 2 AT XX,YY -
     3
990  GOSUB 1220
1000  GOTO 640
1010  HOME : VTAB 21
1020  IF YY - 30 * Q < 10 THEN  PRINT
     "YOU JUST LEFT THE FOREST,TR
     Y AGAIN": XDRAW 2 AT XX,YY -
     3: GOTO 650
1030  FOR N = YY TO YY - 30 * Q STEP
      - 1
1040  XDRAW 2 AT XX,N - 3: FOR A =
     1 TO 25: NEXT A: XDRAW 2 AT
     XX,N - 3: NEXT N
1050 YY = YY - 30 * Q
1060  HCOLOR= 0: SCALE= 2: DRAW 3
      AT XX,YY
1070  SCALE= 1
1080  HCOLOR= 3: XDRAW 2 AT XX,YY
```

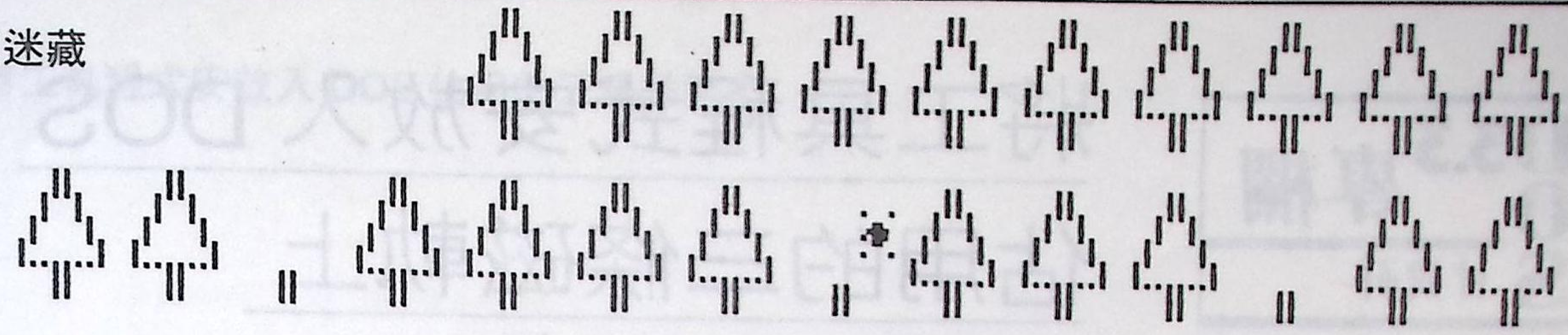
捉迷藏



```
       - 3: SCALE= 1
1090  GOSUB 1220
1100  GOTO 640
1110  REM
1120  HOME : VTAB 21
1130  IF YY + 30 * Q > 130 THEN  PRINT
     "YOU JUST LEFT THE FOREST,TR
     Y AGAIN": XDRAW 2 AT XX,YY -
     3: GOTO 650
1140  FOR N = YY TO YY + 30 * Q
1150  XDRAW 2 AT XX,N - 3: FOR A =
     1 TO 25: NEXT A: XDRAW 2 AT
     XX,N - 3: NEXT N
1160 YY = YY + 30 * Q
1170  HCOLOR= 0: SCALE= 2: DRAW 3
      AT XX,YY
1180  SCALE= 1
1190  HCOLOR= 3: XDRAW 2 AT XX,YY
      - 3: SCALE= 1
1200  GOSUB 1220
1210  GOTO 640
1220  REM
1230  IF XX = X AND YY = Y THEN  SCALE= 1:
     HCOLOR= 3: DRAW 1 AT XX,YY: GOTO 1300
1240 B = X - XX: IF B < 0 THEN B =
     B * ( - 1)
1250 C = Y - YY: IF C < 0 THEN C =
     C * ( - 1)
1260 T =  SQR (B ^ 2 + C ^ 2)
1270  IF T > 255 THEN T = 255
1280  GOSUB 1740
1290  RETURN
1300  HOME : VTAB 12
1310  GOSUB 1380
1320  TEXT : HOME : VTAB 3: FOR I
      = 1 TO 19: INVERSE : PRINT
     "*";: FLASH : PRINT "*";: NEXT
     I: HTAB 1: VTAB 22: FOR I =
     1 TO 19: INVERSE : PRINT "*"
     ;: FLASH : PRINT "*";: NEXT
     I
1330  VTAB 10: HTAB 11: FLASH : PRINT
     "YOU ARE THE WINNER": NORMAL
     : PRINT : PRINT
1340  PRINT "YOU TOOK ";L;" GUESS
     ES TO FIND THE LION"
1350  PRINT "DO YOU WANT TO PLAY
     AGAIN? Y/N": GET W$: IF W$ <
      > "N" THEN  GOTO 50
1360 D$ =  CHR$ (4)
1370  PRINT : PRINT D$"CATALOG": END

1380  FOR N = 1 TO 20
1390  POKE 768,60: POKE 769,60
1400  CALL 770
1410  POKE 768,180: POKE 769,60
1420  CALL 770
1430  NEXT N
1440  RETURN
1450  HOME : VTAB 21:L = L - 1:M =
     0
1460  PRINT "I DECIDED TO HIDE BE
     HIND ANOTHER TREE ."
1470  PRINT "HA HA HA HA ....."
1480  FOR N = 1 TO 10: GOSUB 1710
     : NEXT N
1490  INPUT "IF YOU WANT TO FIND
     ME PUSH RETURN";QQ$
1500 SS = 14
1510  RETURN
1520  CALL 62450
1530  SCALE= 1
1540  FOR GG = 1 TO 5
1550  XDRAW 1 AT 110,80
1560  FOR N = 1 TO 90:Z =  PEEK (
      - 16336): NEXT N
1570  HOME
1580  XDRAW 1 AT 110,80
1590  SCALE= GG * 3
1600  NEXT GG
1610  HOME : TEXT
1620  VTAB 12: HTAB 10: PRINT "I
     ATE YOU ALL UP..YUMMY"
1630  VTAB 21: GOTO 1350
1640  END
1650  PRINT
1660  POKE 770,173: POKE 771,48: POKE
     772,192: POKE 773,136: POKE
     774,208: POKE 775,5: POKE 77
     6,206: POKE 777,1: POKE 778,
     3: POKE 779,240: POKE 780,9:
      POKE 781,202
1670  POKE 782,208: POKE 783,245:
      POKE 784,174: POKE 785,0: POKE
     786,3: POKE 787,76: POKE 788
     ,2: POKE 789,3: POKE 790,96:
      POKE 791,0: POKE 792,0
1680  RETURN
1690  POKE 768,Y: POKE 769,30
1700  CALL 770
1710  POKE 768,200: POKE 769,30
1720  CALL 770
1730  RETURN
1740  REM
1750  POKE 768,T: POKE 769,255
1760  FOR N = 0 TO 4
1770  CALL 770
1780  NEXT N
1790  RETURN
1800  RESTORE
1810  READ I,J: IF J = 0 THEN  RETURN

1820  POKE 768,I: POKE 769,J: CALL
     770
1830  GOTO 1810
1840  DATA  128,120,152,60,114,6
     0,128,120,152,60,00,00
1850  FOR N = 1 TO 1
```



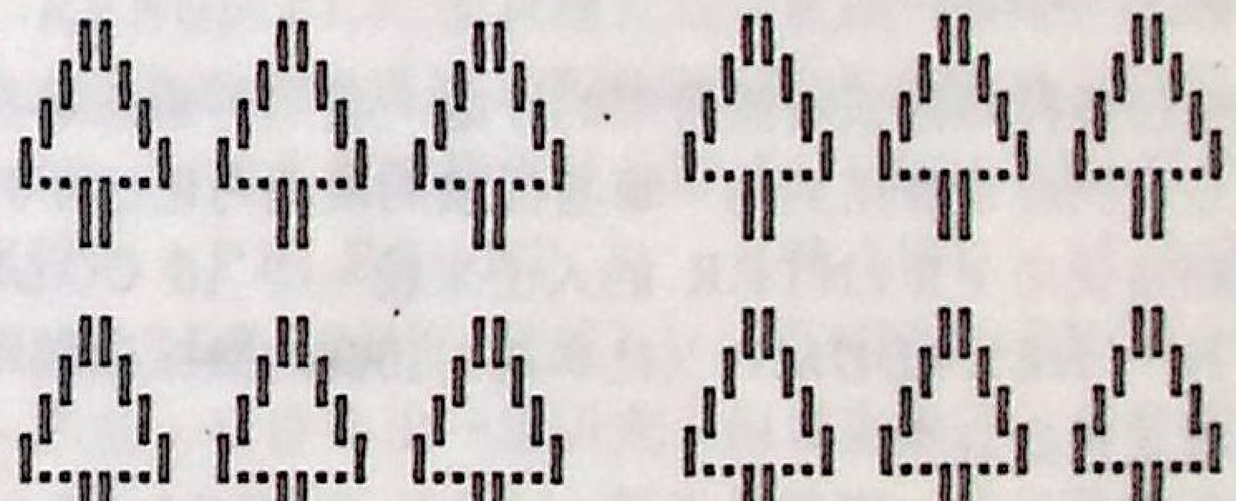
```
1860  POKE 768,255: POKE 769,60
1870  CALL 770
1880  NEXT N
1890  RETURN
1900  HOME : VTAB 12: PRINT "DO W
    ANT INSTRUCTIONS? Y/N": INPUT
    AA$: IF AA$ < > "Y" THEN  RETURN

1910  HOME : VTAB 3: HTAB 14: FLASH
    : PRINT "LION HUNTER ": PRINT
    : PRINT
1920  NORMAL
1930  PRINT "LION HUNTER  IS A GA
    ME WHICH USES"
1940  PRINT "ANIMATION AND SOUND
    TO INTRIGUE YOUR"
1950  PRINT "KIDS WHILE TEACHING
    THEM DIRECTION AND "
1960  PRINT "LOGIC THROUGH LISTEN
    ING,LOOKING,AND"
1970  PRINT "COUNTING."
1980  PRINT "THE OBJECT OF THE GA
    ME IS TO FIND THE"
1990  PRINT "LION AFTER HE LANDS
    IN THE FOREST ."
2000  PRINT "IF IS NOT FOUND WITH
    IN 10 GUESSES,"
2010  PRINT "THE LION WINS."
2020  PRINT "STARTING AT THE TOP
    LEFT CORNER OF THE"
2030  PRINT "SCREEN,THE CHILD CAN
    MOVE UP,DOWN,"
2040  PRINT "LEFT OR RIGHT BY PRE
    SSING U,D,R, OR L"
2050  PRINT "AND A NUMBER WHICH S
    IGNIFIES HOW MANY"
2060  PRINT "TREES HE WISHES TO M
    OVE."
2070  PRINT : PRINT "PRESS ANY KE
    Y TO CONTINUE...": GET K$: HOME
    : VTAB 3: HTAB 14: FLASH : PRINT
    "LION HUNTER ": PRINT : PRINT

2080  NORMAL
2090  VTAB 10
2100  PRINT "YOU MAY CHOOSE THE
    EASIER' VERSION"
2110  PRINT "(WHICH ALLOWS THE LI
    ON TO PEEK OUT"
2120  PRINT "FROM HIS HIDING PLAC
    E AFTER EACH MOVE)"
2130  PRINT "FOR YOUR CHILD BY EN
    TERING HIS NAME"
2140  PRINT "BELOW.(JUST HIT RETU
    RN IF NONE)"
2150  PRINT "THE EASIER VERSION W
    ILL AUTOMATICALLY"
2160  PRINT "RUN WHEN THE CHILD B
    ELOW PLAYS THE GAME."
2170  PRINT "ANY OTHER KIDS WILL
    AUTOMATICALLY PLAY"
2180  PRINT "THE REGULAR VERSION
    ."
2190  VTAB 22: INPUT "CHILD'S NAM
    E...";J$: HOME : VTAB 12: HTAB
    5: PRINT "HIT ANY KEY TO BEG
    IN ..": GET B$
2200  RETURN
2210  END
```

```
BLOAD LION
*5600.5680

5600- 04 00 15 00 54 00 6A 00
5608- 7B 00 00 00 00 00 00 00
5610- 00 00 00 00 00 36 36 2E
5618- 2D 2D 2D 25 24 24 2D 2D
5620- 36 36 36 36 2D 24 24 24
5628- 24 24 3F 3F 3F 36 1B 64
5630- 21 3C 3F 3F 27 2C 2D 35
5638- DE 1B 3F 36 3F 3F 3F 36
5640- 36 36 36 36 2D 24 24 24
5648- 24 2D 2D 64 31 56 11 35
5650- 3E 27 2C 00 2C 2E 25 24
5658- 36 36 36 26 24 3F 3E 3C
5660- 37 36 24 24 24 34 36 2E
5668- 2C 00 09 2E 36 3F 3F 3F
5670- 27 2C 24 25 2C 24 35 2E
5678- 36 35 00 92 36 3E 24 04
5680- 00
*_
```



DOS 3.3 專欄

甘貴祥

將工具程式安放入DOS佔用的三條磁軌上

我們在日常使用電腦時，若把一些喜愛的工具程式載入記憶體內，在 BOOT 機之後都必須經過再次讀取磁碟，撇開所需花於讀取的時間不理，單是磁碟上空間便會被浪費不少，假如你有三個工具程式需要加進二十隻磁碟上，而每個程式長度以 3 K BYTES 為例，每隻磁碟便需佔有 3 × 4 ＋ 1 ＝13 SECTORS，二十隻磁碟要耗費 260 個 SECTORS 。一隻經格式化磁碟會有 560個SECTORS，可用的有496 個 SECTORS，換言之，我們可以說是浪費了龐大的記憶空間，其實這些被浪費的空間是有方法可節省的。

本程式便是針對在上面所列出的兩個大前題，而設計出的程式，其優點：

① 避免每次在載入這些工具程式時，要額外花費時間。
② 將程式存放在 DOS 所佔用的 3 條磁軌上，故不會減少磁碟使用空間。
③ 可以加入任何自己喜愛的工具程式入 DOS 內，而只佔原來磁碟上第 0 ， 1 ， 2 三條磁軌。
④ 這種經過擴展的 DOS 能夠將已存有資料的磁碟格式化（ INITIALIZE ），而不會損壞已存在的資料。
⑤ 除 DOS 3.1外，任何其他 DOS 也可適用於本程式。

以上五點功能，分別由兩個程式來完成，第一個是 DOS EXTEND ，它可做第①至③項，而第二個 SLAVE INITIALIZER 是在 DOS EXTEND 完成①至③項後，由它來完成第④及⑤項。

使用範圍

要充份發揮本程式所能帶來的好處，我們可想想那些工具程式是我們經常會用到的，筆者提議的範圍可包括下列：

① 對經常使用 PRINTER 的人士來說，將 16 COLUMN HEX•DUMP （見本期的16欄的機械碼編印）等程式包括在內。
② 對經常設計程式的人士來說，可將 SYSTEM MASTER上的 RENUMBER存放在內，或存一個高解像度文字產生器等都會帶來很多方便。

使用需知

① DOS EXTEND必須在載入機械語言程式往 DOS 內之前執行。對 APPLESOFT 程式則須改動一些指向器數值，技術上來說是採用將 APPLESOFT 與機械語言合併的技術來達成，詳請可參考同期的〔混合機械語言／ APPLESOFT程式的技術初探〕一文。
② 計算出所需的空間， 256 BYTES的記憶頁或 595 BYTE 的檔案緩衝器之倍數，以 DOS EXTEND 程式將這個長度設定，再以 BLOAD 命令載入機械語程式，經核對無誤後再用 SLAVE INITILIZER 將磁碟重新格式化。

程式的限制

DOS EXTEND一經執行後，會將 HIMEM 及 MAX FILES 的數值更改，因此除非經過修改否則有些程式不能運行 。

執行原理

APPLE Ⅱ 的 BASIC INTERPRETER 原本是被設計成可在不同的使用空間下運作，這個空間由 4 K至48 K不等，當開機時，它會首先決定使用空間有多少，並且將這個數值存入 HIMEM 之內（在 INTEGER BASIC 是 $4C及 $4D，在 APPLESOFT 為 $73 及 $ 74 上），使BASIC INTERPRETER 不會佔用比 HIMEM 數值更高的記憶空間，避免破壞記憶體內 HIMEM 位址以上的內容。不過就算你設定 HIMEM 後使用過 FP，INT， 或 MAXFILE 這三個命令，都會令 HIMEM 重置回 $ 9600 去。為使各位易於理解，請先參考表一。

位 址	功能
$BFFF	RWTS（讀取或寫出資料到磁碟的 TRACK 及 SECTOR）。
$B600	磁碟存取的初步執行網路。
$B5FF	FILE MANAGER 磁碟上檔案命令網路。
$AAC1	呼叫 RWTE 網路之用。
$AAC0	操作系統。
$9000	（包括 BASIC 介面的命令翻譯器，並呼叫 FILE MANAGER）。
$9CFF	檔案緩衝器並設定 MAXFILES＝3。
$9600	
$95FF	DOS 將 HIMEM 設定於這個位址＋1的數字，APPLESOFT 的字串變數及 INTEGER BASIC 由這裡開始向下存放。

表(一)

要將機械語言程式存放入DOS 之內，而不管按入 FP ，INT 或 MAXFILES 都不會變更 HIMEM 的方法是將這些程式放在 $9D00的 INTERPRETER之下，及在檔案緩衝器之上，這樣一來就算怎麼樣改變檔案緩衝器（包括 BOOT 機， FP，INT 及 MAXFILES）亦不會破壞機械語言程式。

進行的方法是改變（ 40192－40193／$9D00，9D01）指內器的值，使它指向（ 42964／$A7D4）另一指向器上，這指向器在每個檔案緩衝器上會佔用為首的 38 個 BYTE ，故此我們須至少改變它到需加入的程式之下的空間 38個 BYTE 之位置去，例如要保留 $9C80及9CFF 作為一個機械語言程式的存放之用，那末必須將指向器改為 $9C89－$26＝$9C5A 或更低的數值。存放在$9D00 這個指向器之內，以低位元為首 9D00：599C

將工具程式安放入DOS佔用的三條磁軌上

〔RETURN〕，之後執行 $A7D4 G〔RETURN〕使檔案緩衝器及另外數個指向器的值改變爲合適的值。

通常這個指向器會設定在DOS 解釋器之下 45〔38＋7＝45（$2D）〕個 BYTE，因此以三個595（$293）BYTE的檔案長度作爲間隔，使HIMEM 下降至1792（595×3＋7＝1792＝256×7）的單數檔案緩衝器邊界去，此時 HIMEM 會被設定爲 $9600 而非 $9607），從以上的原理出發，最好以雙數乘以256（$100）個 BYTE 來擴展DOS 的空間。

其次本程式可以用檔案緩衝器 595（$293）BYTE 或其倍數來擴展 DOS，若你將 MAXFILE的值降低，HIMEM 仍會不變，MAXFILE 是在 43607／$AA57 上，而其自設值則存放在43697／$AAB1，這自設值是當 BOOT 機，RESET，FP 或 INT 被啓動時自定的數值。將 HIMEM 維持在 $9600，可令擴展後的 DOS操作正常。

綜合所需要的情形，DOS EXTEND 被設計成：

① 以 256 BYTE 的倍數來延伸程式空間；
② 以 595 BYTE 的檔案緩衝器或其倍數來延伸程式空間；
③ 以任何任意的數值來延長記憶空間；
④ 可以更改 MAXFILE的自設值，而維持HIMEM 在原來位置。

在未存有資料的磁碟上擴展 DOS

若記憶體中存有已被擴展的 DOS（包括機械語言程式在內）。以 INIT 命令來格式化一隻新磁碟，該新磁碟的 DOS 將會同樣是擴展了的DOS ，亦即包括機械語言程式在內。

對已存有資料的磁碟上擴展 DOS

我們知道以上的方法會把已存有資料的磁碟上任何東西抹去，因此必須以 SLAVE INITILIZER 來處理此一問題，SLAVE INITILIZER 能夠將已擴展的 DOS 寫回磁碟上，而不會破壞原有的檔案。

要使 SLAVE INITILIZER運行無誤必須達到以下其中一項要求：

① 已被擴展的DOS 影像存在於記憶體或記憶體中或；
② 從 DOS 3.3 MASTER SYSTEM 中取出 MAST—ER DOS 3.3 IMAGE，而經執行 DOS EXT—END 使該 DOS 被擴展，及加入其他機械語之程式後再執行 SLAVE INITILIZER使此 DOS 存在已存有資料的磁碟上。
③ 經由上述第②點所製成之 DOS，經關掉及再啓動電源而載入記憶體內之DOS·擴展版本。

符合以上①，②，③項其中之一時，磁碟上的資料則基本上確保不會因意外而被破壞。

如何確定 SLAVE INITIALIZER 有效

本程式可以把任何記憶體內被視爲有效的 DOS 放進磁碟去，另一方可以更改 GREETING PROGRAM 的名字，再把新名字連同記憶體內的 DOS 影像寫回磁碟。

執行本程式時，首先會在螢幕上顯示現時的 DOS 版本及記憶體的大小，跟着是要輸入 GPEETING PROGR—AM 的名字，程式會指示你插入磁碟再按 RETURN，數秒後便會完成重寫入的工作。

完成以上步驟後，將電源關掉再開，檢查剛才寫入磁碟上的 DOS 能否再次 BOOT 入記憶體，並檢查有關命令是否正常（以上步驟對已擁有本刊程式磁碟之人士來說是無須理會的）。

怎樣對磁碟作 DOS EXTEND

在實際的使用上，DOS EXTEND 可以用來檢查記憶體的 HIMEM 值，以及檢查記憶體內的 DOS 影像是否已被擴展過，MAXFILE 的數值等，省却很多時間去找這些資料。不過最初使用時，需要注意一些使用技術。以下我將用實際例子說明。

例①：將 MASTER SYSTEM上的 RENUMBER 程式改良及放置於 DOS 內。

RENUMBER 工具程式過去曾經一度受到歡迎，尤其在程式編寫設計時爲然，不過閣下有否體驗過，當 RENU—MBER 程式在記憶體內而執行 INITEGER BASIC 後再回到 APPLESOFT 時，（按 INT，或 LOAD 入一個 APPLESOFT 程式），RENUMBER 便會失去作用，因此，我曾作出一番研究，但都失敗了。最後我看到 CALL APPLE 1981 年7月號的一篇文章討論有關 R—ENUMBER 的轉移位址程式內存有 BUG，才如夢初醒，終於將問題解決，方法如下：

LOAD RENUMBER
CALL—151按 B62：4C D0 0B〔RETURN〕
BD0：C9 D6 D0 03 4C 70 0B 18 69 10 4C 65 0B〔RETURN〕 1085：D6〔RET—URN〕

第二個 BUG 是 RENUMBER一旦 LIST過時；便不能正常工作，解決方法如下：

CALL—151 12B5：AC A B〔RETURN〕這時可以按 CTRL—C 回到 BASIC去，再 SAVE CORR—ECT RENUMBER。

到現時我們要製造這個 RENUMBER DOS了，這樣才能在 BOOT 機時令 RENUMBER長駐記憶之內。方法

將工具程式安放入DOS佔用的三條磁軌上

是 RUN DOS EXTEND ，程式會顯示出DOS 的版本及記憶體的大小，現時及一般情況下第一個檔案緩衝器＋7的位址，這兩個數字一般都會相同，在48K的 APPLE DOS 3.3 下是 ＄9D00，不過若果DOS 經過修改的話又作別論。這時你應該記下此一位址以後需要之用，除此以外本程式會顯示現時及自設值的 MAXFILE 數字。電腦跟着會詢問閣下，希望以何種形式擴展 DOS ，如圖1所示，按入（1）即以 256 BYTE 的整個倍數來擴展，電腦會再問需要擴展多少個 SECTOR，這時按入9，因爲 RENUMBER 的長度是2048個 BYTES。

電腦這時會顯示出檔案緩衝器被重建在 DOS 之下的有關數字，如圖2所示。現在電腦會問你是否要把已擴展的DOS 存回磁碟中，回答"Y"，程式會顯示 DOS 會擴展9個 SECTORS 來容納要待加入 DOS 的程式，並問你所需擴展的總數，回答9。至此 DOS EXTEND已告使用完畢，下一步是將 CORRECT RENUMBER從磁碟上載入，再按 CALL－151 進入監督程式，鍵入73：00 9D〔RETURN〕(這個數字是你剛才抄下的)。現在你可以按RUN ，RENUMBER 已移往＄9500 亦即擴展DOS之內了。

因爲 RENUMBER 是用 ＄3F5 位址的&字來啓動的，因此，進入監督程式按入 3F5：4C 0095〔RETURN〕，9E76：4C 0095〔RETURN〕，這次一切步驟已完成，只要用 INIT 命令來格式化一隻新磁碟，新DOS 便會永遠保留起來，在 BOOT 機時，RETURN 會長駐記憶體中，另外若使用 SLAVE INITILIZER 把現時已擴展的DOS 寫到已存有資料的磁碟上，則亦可收到同樣效果。

本文共有兩個程式，它們在程式磁碟上的 FILENAME 是 DOS EXTEND 及 SLAVE IMITILIZER。□

```
EXTEND DOS IN MEMORY BY:
1:AN INTEGER MULTIPLE OF 256 BYTES.
2:AN INTEGER MULTIPLE OF 595 BYTES.
  (EQUIVALENT TO ONE FILE BUFER).
3:AN ARBITRARY NUMBER OF BYTES.
4:QUIT.

NOTE:EXTENSIONS ARE FROM THE NORMAL
FIRST FILE BUFFER,AND DO NOT INCLUDE
EXTENSIONS ALREADY MADE ,IF ANY.
```

圖(1)

圖(2)

```
EXTEND BY HOW MANY 256 BYTE SECTORS?9

FILE BUFFERS HAVE BEEN REBUILT BENEATH

MEMORY EXTEND DOS.

CURRENT DOS:48K DOS 3.3

TOP OF FIRST FILE BUFFER (+7):

CURRENT:37888/$9400  NORMAL:40192/$9D00
(LOCATE USER ROUTINES IN ABOVE RANGE.)

CURRENT MAXFILES: 3  DEFAULT: 3

EXTEND DOS FOR SAVING TO DISK(Y/N)?Y
```

```
DOS BOTTOM  FOR INIT:

CURRENT:40192/$9D00  NORMAL:40192/$9D00
(ROUTINES IN THE ABOVE RANGE WILL BE
WRITEN TO DISK WITH DOS DURING IN IT.)

DOS SHOULD BE EXTENDED BY9SECTORS TO
INCLUDE ALL OF MEMORY EXTENDED DOS .

EXTEND DOS BY HOW MANY SECTORS TOTAL?
(0-13)9

DOS HAS BEEN EXTENDED FOR DISK SAVE

(THROUGH INIT OR SLAVE MAKER).

CURRENT DOS:48K DOS 3.3

TOP OF FIRST FILE BUFFER (+7):

CURRENT:37888/$9400  NORMAL:40192/$9D00
(LOCATE USER ROUTINES IN ABOVE RANGE.)

CURRENT MAXFILES: 3  DEFAULT: 3

DOS BOTTOM  FOR INIT:

CURRENT:37888/$9400  NORMAL:40192/$9D00
(ROUTINES IN THE ABOVE RANGE WILL BE
WRITEN TO DISK WITH DOS DURING IN IT.)
```

圖(3)

將工具程式安放入DOS佔用的三條磁軌上

```
10  REM   ********************
20  REM   * SLAVE INITIALIZER*
30  REM   *  BY ALBERT NG    *
40  REM   *                  *
50  REM   * COPYRIGHT (C)1984*
60  REM   * BY COMPUTING AGE *
65  REM   *                  *
70  REM   ********************
100 TEXT : HOME
110 INVERSE : PRINT "  SLAVE INI
    TIALIZER BY ALBERT NG
    ": NORMAL
120 PRINT "--------------------
    ------------------"
130 FLASH : PRINT "  COPYRIGHT (
    C) 1984 BY COMPUTING AGE  ":
     NORMAL
140 PRINT : PRINT "THIS PROGRAM
    WRITES MEMORY RESIDENT"
150 PRINT : PRINT "DOS TO DISK.(
    SOMETHING LIKE'MASTER"
160 PRINT : PRINT "CREATE'OR 'UP
    DATE 3.2.1.')IT WILL"
170 PRINT : PRINT "WORK WITH ANY
     MEMORY SIZE REGULAR,"
180 PRINT : PRINT "PATCHED,OR EX
    TENDED DOS 3.2(.1) OR 3.3."
190 DA =  PEEK (981) * 256
200 DV =  PEEK (DA + 645) +  PEEK
     (DA + 8547):DV = (DV = 4) +
    2 * (DV = 5) + 2.1 * (DV = 6
    ) + 3 * (DV = 12)
210 PRINT : PRINT "CURRENT DOS:"
    (DA + 8960) / 1024"K DOS 3."
    DV
220 IF DV < 2 THEN  INVERSE : PRINT
    : PRINT "THIS VERSION OF DOS
     NOT SUPPORTED!" CHR$ (7); CHR$
    (7); CHR$ (7): NORMAL : END
230 POKE 34, PEEK (37) + 1
240 HOME : PRINT "ENTER GREETING
     PROGRAM NAME:"
250 PRINT : INPUT "";F$
260 F$ =  LEFT$ (F$ + "
                    ",30)
270 VTAB  PEEK (37): PRINT F$",R
    IGHT?";
280 PRINT  CHR$ (7);: GET G$: ON
    G$ = "N" GOTO 240: ON G$ <  >
    "Y" GOTO 280
290 POKE 34, PEEK (37) + 1
300 HOME : PRINT "INSERT DISK TO
     BE 'SLAVED',AND THE N"
310 PRINT : INPUT "PRESS THE <RE
    TURN >KEY.";G$
320 POKE DA + 3423,0
330 FOR I = 1 TO 30
340 POKE DA + 3444 + I, ASC ( MID$
    (F$,I)) + 128
350 NEXT I
360 POKE DA + 6891,0
370 POKE DA + 6896,0: POKE DA +
    6897, PEEK (DA + 13)
380 CALL DA + 6730
390 HOME : PRINT "DOS HAS BEEN W
    RITTEN TO DISK ."
400 POKE 34,0     □
```

```
10  REM  ***********************
    **
20  REM  *
     *
30  REM  *      DOS EXTEND
     *
40  REM  *       BY COMPUHOUSE
     *
50  REM  *
     *
60  REM  ***********************
    **
70  TEXT : HOME
80  INVERSE : PRINT " DOS EXTEND
     BY COMPUTING AGE (C) 1984 ":
     NORMAL
90  PRINT "--------------------
    -------------------"
100 MAX = 65535:P16 = 2
110 DEF  FN DIV(A) =  SGN (A) *
     INT ( ABS (A) / (16 ^ P16))
120 DEF  FN MOD(A) = A - 16 ^ P1
    6 *  FN DIV(A)
130 DEF  FN PK2(A) =  PEEK (A) +
    256 *  PEEK (A + 1)
140 DEF  FN DEC(HEX) = HEX - 55 +
    7 * (HEX < 58)
150 DA =  PEEK (981) * 256
160 DV =  PEEK (DA + 645) +  PEEK
     (DA + 8547):DV = (DV = 4) +
    2 * (DV = 5) + 2.1 * (DV = 6
    ) + 3 * (DV = 12)
170 POKE 34,3: HOME
180 PRINT : PRINT "THIS PROGRAM
    EXTENDS DOS IN MEMORY,"
190 PRINT : PRINT "CREATING SPAC
    E FOR MACHINE LANGUAGE"
200 PRINT : PRINT "ROUTINES BETW
    EEN THE BOTTOM OF DOS AND"
210 PRINT : PRINT "THE TOP OF TH
    E FILE BUFFERS.WORKS"
```

將工具程式安放入DOS佔用的三條磁軌上

```
220  PRINT : PRINT "WITH DOS 3.2,
     3.2.1 OR 3.3"
230  GOSUB 920
240  IF DA - 45 <  >  FN PK2(DA) THEN
      INVERSE : PRINT : PRINT "NO
     TE:DOS HAS ALREADY BEEN EXTE
     NED!"; CHR$ (7); CHR$ (7); CHR$
     (7); CHR$ (7): NORMAL
250  VTAB 24: PRINT "PRESS ANY KE
     Y TO CONTINUE.";: GET G$: HTAB
     1: CALL  - 868
260  PRINT : PRINT : PRINT : PRINT
     : PRINT : PRINT : PRINT : PRINT
     : PRINT : PRINT : POKE 34,12
270  PRINT "EXTEND DOS IN MEMORY
     BY:"
280  PRINT "1:AN INTEGER MULTIPLE
      OF 256 BYTES."
290  PRINT "2:AN INTEGER MULTIPLE
      OF 595 BYTES."
300  PRINT "  (EQUIVALENT TO ONE
     FILE BUFER)."
310  PRINT "3:AN ARBITRARY NUMBER
      OF BYTES."
320  PRINT "4:QUIT."
330  PRINT : PRINT "NOTE:EXTENSIO
     NS ARE FROM THE NORMAL "
340  PRINT "FIRST FILE BUFFER,AND
      DO NOT INCLUDE"
350  PRINT "EXTENSIONS ALREADY MA
     DE,IF ANY."
360  PRINT : PRINT "WHICH:";
370  PRINT  CHR$ (7);: GET G$:G =
      VAL (G$): ON G < 1 OR G > 4
      GOTO 370: PRINT G$: PRINT
380  PRINT : PRINT : PRINT : PRINT
     : PRINT : POKE 34,16: HOME
390  ON G GOTO 400,420,460,710
400  INPUT "EXTEND BY HOW MANY 25
     6 BYTE SECTORS?";NS
410 NB = NS * 256: GOTO 470
420  INPUT "EXTEND BY HOW MANY 59
     5 BYTE BUFFERS?";NB
430  PRINT : PRINT "CHANGE MAXFIL
     ES DEFAULT OF" PEEK (DA + 35
     05);: GOSUB 900: IF NO GOTO
     450
440  PRINT  CHR$ (7): INPUT "NEW
     MAXFFILES DEFAULT:(1-16)";X:
      ON X < 1 OR X > 16 GOTO 440
     : POKE DA + 3505,X: GOTO 430
450 NB = NB * 595: GOTO 470
460  PRINT "EXTEND BY HOW MANY BY
     TES": GOSUB 720:NB = DP: PRINT
     "-RIGHT";: GOSUB 900: IF NO GOTO
     460
470  POKE DA, FN MOD(DA - 45 - NB
     ): POKE DA + 1, FN DIV(DA -
     45 - NB)
480  POKE DA + 3415, PEEK (DA + 3
     505)
490  CALL DA + 2772
500  POKE 34,4: HOME
510  PRINT : PRINT "FILE BUFFERS
     HAVE BEEN REBUILT BENEATH": PRINT
     : PRINT "MEMORY EXTEND DOS."
520  GOSUB 920
530  PRINT : PRINT "EXTEND DOS FO
     R SAVING TO DISK";: GOSUB 90
     0: IF NO GOTO 710
540  PRINT : PRINT : PRINT : PRINT
     : PRINT : PRINT : PRINT : PRINT
     : PRINT : POKE 34,10: HOME
550  GOSUB 980
560  PRINT : PRINT "DOS SHOULD BE
      EXTENDED BY" FN DIV(DA) -  FN
     DIV( FN PK2(DA) + 45)"SECTOR
     S TO": PRINT "INCLUDE ALL OF
      MEMORY EXTENDED DOS ."
570  IF  FN PK2(DA + 12) <  > (DA
     ) THEN  INVERSE : PRINT : PRINT
     "DOS INIT BOTTOM  HAS ALREAD
     Y BEEN": PRINT "EXTENDED BY
     " FN DIV(DA) -  PEEK (DA + 1
     3)" SECTORS.": NORMAL
580  PRINT : PRINT "EXTEND DOS BY
      HOW MANY SECTORS TOTAL?"
590  PRINT "(0-"4 + 9 * (DV = 3);
     : PRINT  CHR$ (7);: INPUT ")
     ";NS: ON NS < 0 OR NS > 4 +
     9 * (DV = 3) GOTO 590
600  POKE DA + 13, FN DIV(DA) - N
     S
610  IF NS > 11 THEN NS = 11
620  IF DV <  > 3 THEN 660
630  POKE DA + 6682,4 + NS
640  POKE DA + 6755,4 + NS
650 NS = NS + 2
660  POKE DA + 6880,25 + NS
670  POKE 34,3: HOME
680  PRINT : PRINT "DOS HAS BEEN
     EXTENDED FOR DISK SAVE": PRINT
     : PRINT "(THROUGH INIT OR SL
     AVE MAKER)."
690  GOSUB 920
700  GOSUB 980
```

將工具程式安放入DOS佔用的三條磁軌上

```
710  POKE 34,0: END
720  INPUT "($ INDICATES HEXADECI
     MAL):";G$
730 DP =  VAL (G$)
740  IF  LEFT$ (G$,1) <  > "$" GOTO
     800
750 DP = 0
760 HX$ =  RIGHT$ ("0000" +  MID$
     (G$,2),4)
770  FOR P16 = 3 TO 0 STEP  - 1
780 DP = DP * 16 +  FN DEC( ASC (
     MID$ (HX$,4 - P16,1)))
790  NEXT P16:P16 = 2
800  IF  ABS (DP) > MAX THEN 720
810  IF DP < 0 THEN DP = DP + MAX
820 N = DP
830 HX$ = ""
840  FOR P16 = 3 TO 0 STEP  - 1
850 HX$ = HX$ +  MID$ ("012345678
     9ABCDEF",  FN DIV(N) + 1,1)
860 N =  FN MOD(N)
870  NEXT P16:P16 = 2
880  PRINT DP"/$"HX$;
890  RETURN
900  PRINT "(Y/N)?";
910  PRINT  CHR$ (7);: GET G$:YES
     = G$ = "Y":NO = G$ = "N": ON
     NOT (YES OR NO) GOTO 910: PRINT
     G$: RETURN
920  PRINT : PRINT "CURRENT DOS:"
     (DA + 8960) / 1024"K DOS 3."
     DV
930  IF DV < 2 THEN  INVERSE : PRINT
     : PRINT "THIS VERSION OF DOS
      NOT SUPPORTED!" CHR$ (7); CHR$
     (7); CHR$ (7): NORMAL : GOTO
     710
940  PRINT : PRINT "TOP OF FIRST
     FILE BUFFER (+7):"
950  PRINT : PRINT "CURRENT:";:DP
     =  FN PK2(DA) + 45: GOSUB 8
     20: PRINT "  NORMAL:";:DP =
     DA: GOSUB 820: PRINT :  IF G THEN
      PRINT "(LOCATE USER ROUTINE
     S IN ABOVE RANGE.)"
960  PRINT : PRINT "CURRENT MAXFI
     LES: " PEEK (DA + 3415)"  DE
     FAULT:  "; PEEK (DA + 3505)
970  RETURN
980  PRINT : PRINT "DOS BOTTOM  F
     OR INIT:"
990  PRINT : PRINT "CURRENT:";:DP
     =  FN PK2(DA + 12): GOSUB 8
     20: PRINT "  NORMAL:";:DP =
     DA: GOSUB 820: PRINT : PRINT
     "(ROUTINES IN THE ABOVE RANG
     E WILL BE": PRINT "WRITEN TO
      DISK WITH DOS DURING IN IT.
     )"
1000  RETURN □
```

新DOS命令改換程式

達

讀者中可有想過用一些簡單的命令來取代冗長的 DOS 命令？雖然本刊以前也提供了兩個這類的 DOS 命令轉換程式，惟筆者仍覺得他們有未盡善之處，如在第六期的程式，改了的命令不會顯示出來，便令用家在較後的日子裏不知道實際的 DOS 命令，本程式就是吸收了它的優點，再把缺點加以改良，而寫成了今次爲各位介紹的新 DOS 命令改換程式。

程式的操作

本程式可以把原來的 DOS 命令縮短至兩個或三個的字母，因此而減輕了用家在程序鍵入時所產生的錯誤。爲了安全起見，讀者在第一次執行本程式時，可先執行"MASTER CREATE"的 DOS 程式，這樣做是爲了讓磁碟上的命令與正常的 DOS 命令一致的關係。當程式被觸發時，用家會被要求放入正待更換的磁碟，當放了磁碟後，按〔RETURN〕鍵，程式便會讀取磁碟上在第一條磁軌的第 6－7 SECTOR 的28個 DOS 命令，再把它們在螢幕分兩行列出，每一個命令都配予一個號碼，而在命令的右邊，就是標準的 DOS 命令，這樣做法，是令用家可以作一比較及參考。例如要改變"CATALOG"這個命令，用家可按數字"17"，接著打入"CAT"及〔RETURN〕程式便會在第17行中加深了"CAT"的字母作爲標示。當全部改換完成後，按〔RETURN〕便會把新的命令轉移至磁碟上，整個操作便完成。

不過使用本程式時必須要留意以下爲幾個規則用家要遵守，

（1）不要用少於兩個字母的命令；

（2）不要用多於 132 個字母的命令，因爲標準的 DOS 命令的總長爲 132個字母，所以28個命令的總和應以不超過比數爲限；

（3）只可用英文字母，而"#"這個字符號只可以在命令的末端才可應用；

（4）假如用家不遵守以上的規則，DOS 便有極可能被毀滅，要救回DOS，用家可以執行"MA－STER CREATE"的程式；

（5）不要重覆使用相同的命令，DOS 只會執行第一個的命令，而其餘相同的則會被認作錯誤；

（6）RUN 的命令不可以轉換，因爲"RUN"實際上是要經過"LOAD"的步驟才執行的，假如"LOAD"已被改掉，那麼"RUN"便會不知所從。

程式的工作原理

28個 DOS 命令放在第 1 磁軌的 6－7 SECTOR 上，而每一個命令的最後字母爲命令的終端標記（要達致這個效果，它們都加大了 128）。程式把它們全部讀入了矩陣 N$ 中，其邏輯做法可參看行號 1000－1150。於是程式便要求用家鍵入要更改的命令，當輸入完成後，程式便會把結果送入矩陣 N$中（行號 20001－20060），再把 N$轉爲至矩陣×中（行號 20070－20130），而矩陣×則儲存了28個 DOS 命令的 ASCⅡ 字符。最後的步驟就是把新的命令（矩陣×）寫回磁碟上（行號20150－20200），這便完成了整個操作。用家這時可打入"PR#6"重新"BOOT"起 DOS，試試新的 DOS 命令。□

```
1   REM  ****************************

2   REM  **        NEW             **

3   REM  ** DOS COMMAND CHANGER **

4   REM  **                        **

7   REM  ****************************

10  PRINT ""
100  DIM N$(28),X(200),S$(28),NC$
    (28)
110  TEXT : HOME : SPEED= 255
115  REM  LINES 120-124 FIND THE
    DOS VERSION AND SET DS PARAM
    ETER
116  REM  DOS 3.3 --> DS = 7
117  REM  DOS 3.2 --> DS = 8
120 Q =  PEEK (978):Q1 = (Q + 34)
     * 256 + 214:Q2 = (Q + 9) *
    256 + 179:DS = 0
122  IF  NOT  PEEK (Q2) AND ( PEEK
    (Q1) = 240 OR  PEEK (Q1) = 9
    6) THEN DS = 8
124  IF  PEEK (Q2) = 3 THEN DS =
    7
128 KT = 1:Y = 1:BY = 132: IF DS =
    0 THEN  PRINT "I CANNOT WORK
    WITH THIS DOS.": PRINT "THE
    DOS LOADED IS NOT3.2 OR 3.3
    ": END
130  FOR I = 1 TO 28: READ S$(I):
    NEXT
140 T$ = S$(22):S$(22) = S$(8):S$
    (8) = T$
500  IF  PEEK (987) < 120 THEN IO
    B =  PEEK (987) * 256 + 232
510  IF  PEEK (987) > 120 THEN IO
    B = ( PEEK (987) - 128) * 25
    6 + 231 - 32767
520  POKE 1001,76: POKE 1002,181:
    POKE 1003, PEEK (987):RWTS =
    995
530 TRK = IOB + 4:SEC = IOB + 5:A
    HIGH = IOB + 9:ALOW = IOB +
    8:COM = IOB + 12:VOL = IOB +
    3:ERR = IOB + 13
```

```
540  POKE TRK,KT
550  POKE SEC,DS
560  POKE VOL,0
570  POKE COM,1
580  POKE AHIGH,179
590  POKE ALOW,187
600  POKE ERR,0
610  PRINT "INSERT DISK AND PRESS ANY KEY ";: GET U$
620  CALL RWTS
630 LOC = (179 * 256) + 187
640 KP = LOC + BY
650 NLOC = LOC + 256
660  POKE AHIGH, INT (NLOC / 256)
670  POKE ALOW,NLOC - 256 *  INT (NLOC / 256)
680  POKE SEC,DS + 1: CALL RWTS
1000 ST = LOC + BY
1010  FOR K = 1 TO 28
1020  PRINT ".";
1030 X =  PEEK (ST)
1050  IF X > 96 AND X < 192 THEN X = X - 64
1060  IF X > 192 THEN X = X - 128 :FL = 9
1070 C$ =  CHR$ (X)
1080 IN$ = IN$ + C$
1090  IF X = 35 OR X = 99 THEN FL = 9
1100  IF FL = 9 THEN FL = 0: GOTO 1120
1110 ST = ST + 1: GOTO 1030
1120 N$(K) = IN$:IN$ = ""
1130 ST = ST + 1
1140  NEXT K
1150 T$ = N$(22):N$(22) = N$(8):N$(8) = T$
2000  HOME : PRINT  TAB( 15)"COMMAND CHANGER"
2005 IF DS = 8 THEN  VTAB 16: HTAB 16: PRINT "2";: HTAB 1
2010  VTAB 3: PRINT "  CURR"; SPC( 5);"STD"; SPC( 09);"CURR"; SPC( 4);"STD"
2020  FOR I = 1 TO 40: PRINT  CHR$ (45);: NEXT
2030  VTAB 5
2040  FOR I = 1 TO 14
2050  HTAB 1: PRINT I;: HTAB 4: PRINT N$(I);: HTAB 13: PRINT S$(I);: HTAB 22: PRINT I + 14;: HTAB 25: PRINT N$(I + 14);: HTAB 33: PRINT S$(I + 14)
2060  NEXT
2070 T$ = N$(22):N$(22) = N$(8):N$(8) = T$
2080  VTAB 20: HTAB 1: CALL  - 958: INPUT "ENTER SELECTON NUMBER -> ";S$:S =  VAL (S$): IF S < 0 OR S > 28 THEN 2080
2085  IF S = 8 OR S = 22 THEN G = 9
2090  IF S = 0 THEN 20000
2100  IF S > 14 THEN 2140
2110  VTAB 4 + S: HTAB 1: FLASH : PRINT S: NORMAL
2120 U = 9
2130  GOTO 2150
2140  VTAB 4 + S - 14: HTAB 22: FLASH : PRINT S: NORMAL
2150  VTAB 20: HTAB 1: CALL  - 958: INPUT "ENTER NEW COMMAND -> ";NC$: IF NC$ = "" THEN 2150
2160  IF U = 9 THEN U = 0: VTAB 4 + S: HTAB 1: INVERSE : PRINT S: NORMAL : GOTO 2180
2170  VTAB 4 + S - 14: HTAB 22: INVERSE : PRINT S: NORMAL
2180 NC$(S) = NC$
2190  GOSUB 50000
2200  GOTO 2080
6000  DATA INIT,LOAD,SAVE,RUN,CHAIN,DELETE,LOCK,UNLOCK,CLOSE,READ,EXEC,WRITE,POSITION,OPEN,APPEND,RENAME,CATALOG,MON,NOMON,PR#,IN#,MAXFILES,FP,INT,BSAVE,BLOAD,BRUN,VERIFY
20000 Y = 0: VTAB 21: PRINT "COMPUTING"
20001  FOR I = 1 TO 28
20002  IF G = 9 THEN G = 0:T$ = NC$(22):NC$(22) = NC$(8):NC$(8) = T$
20010  IF NC$(I) = "" THEN 20060
20015 N$(I) = NC$(I)
20060  NEXT
20070  FOR I = 1 TO 28
20080 K =  LEN (N$(I))
20090  FOR J = 1 TO K - 1
20100 Y = Y + 1:X(Y) =  ASC ( MID$ (N$(I),J,1))
20110  NEXT J
20120 Y = Y + 1:P =  ASC ( RIGHT$ (N$(I),1)) + 128
20125 X(Y) = P
20130  NEXT I
20140  FOR ZZ = 1 TO Y: POKE KP +
```

新 DOS 命令改換程式

```
    ZZ - 1,X(ZZ): NEXT
20150  POKE AHIGH,179
20160  POKE ALOW,187
20165  POKE COM,2
20166  POKE SEC,DS
20170  CALL RWTS
20180  POKE AHIGH, INT (NLOC / 25
   6)
20190  POKE ALOW,NLOC - 256 * INT
   (NLOC / 256)
20198  POKE SEC,DS + 1
20200  CALL RWTS
20300  VTAB 21: HTAB 1: CALL - 9
   58: PRINT "DONE"
20400  VTAB 22: HTAB 1: CALL - 9
   58: PRINT "MORE (Y/N) ? ";: GET
   YN$: IF YN$ = "Y" THEN RUN
20405  IF YN$ = "N" THEN END
20406  GOTO 20400
50000  IF S > 14 THEN 50050
50010  VTAB 4 + S: HTAB 4: PRINT
   "      ": VTAB 4 + S: HTAB 4
   : PRINT NC$: GOTO 50090
50050  VTAB 4 + S - 14: HTAB 25: PRINT
   "      ": VTAB 4 + S - 14:
   HTAB 25: PRINT NC$
50090  RETURN
```

第八期捉蟲得獎公佈

第八期捉蟲訓練的得獎人是住在上葵涌石蔭路天安樓的鄭偉源君，他將匿藏起來的蟲一一捉出來，並作以下的解釋。

1、程式的第二部份可能不可完全附合要求。亂數函數用的數子（SEED）須大於零才能產生亂數。否則，若數子等於零，產生的數值和以前最後產生的亂數相同；若數子是負數，則產生的亂數也只是某一數值。所以本人提議另加35!

35 IF S < = 0 THEN 30

2、雜誌上的程式，行105的文字〝STORED〞應該是〝SORTED〞。這可能是人爲的錯誤。

3、程式中利用〝POKE 32，20〞來限制印列由何處開始，但這是不健全，有時會做成嚴重後果。當螢幕需要捲動時，問題發生，文字幕有部份被劃花，而程式也會告消。原因是程式只把WINDOW LEFT EDGE（地址32）由1變成21，WINDOW WIDTH（地址33）却仍然保持40。捲動螢幕的副程式的程序是先把一行的資料搬上上一行，最後把行24轉爲空格（存$AO入地址）。大家知道文字幕$400－$7FF不是全用作螢幕顯視，有小部份地址是看不見，却貯存了某些數值。因地址計算錯誤，那些地址就被搬上螢幕，劃花了它。另外因着同樣道理，有些$AO被存入APPLESOFT BASIC貯存區（$800開始），因此程式被清除。爲了保險、安全計，最好在〝POKE32，20〞後加上〝POKE 33，20〞(限制WINDOW WIDTH），問題就可解决。當然在行110的〝POKE 32，0〞要加上 POKE 33，40，使它回復正常狀態。或者，可用〝HTAB 21〞指令代替〝POKE 32，20〞即

105 VTAB 4：HTAB 21：PRINT〝SORTED ARRAY：〞：FOR I＝1 TO N：HTAB 21：PRINT A（I）：NEXT

此處行110可刪除。

4、行40至行60的迴圈可行65的迴圈可合拼卽：

65 VTAB 4：PRINT〝UNSORTED ARRAY：〞：FOR I ＝ 1 TO N：A（I）＝RND（S）：PRINT A（I）：NEXT

這可使程式簡潔，減少記憶用量。

程式更改如下：

```
10 N = 10
30  TEXT : HOME : VTAB 10: INPUT
     "PLEASE  ENTER THE  SEED FOR
      THE RANDOM FUNCTION:";S
35  IF S < = 0 THEN 30
63  HOME
65  VTAB 4: PRINT "UNSORTED ARRAY
    :": FOR I = 1 TO N:A(I) =  RND
    (S): PRINT A(I): NEXT
70  FOR I = 1 TO N - 1
80  FOR J = N - 1 TO I STEP - 1
90  IF A(J) > A(J + 1) THEN Z = A
    (J):A(J) = A(J + 1):A(J + 1)
    = Z
100  NEXT J: NEXT I
105  POKE 32,20: POKE 33,20: VTAB
     4: PRINT "SORTED ARRAY:": FOR
     I = 1 TO N: PRINT A(I): NEXT

110  POKE 32,0: POKE 33,40
```

第 9 期捉蟲訓練問題——編印質數

自從捉蟲問題一欄在第五期開始登出後，每期都收到不少的參加信，由於獎額有限，許多參加者答案雖然優異但難免亦落空。而且由於每期要趕在截稿時間前揀出得獎者，對後來的參加者會有點不公平。編輯部經過會議後，决定由今期開始（第九期）改變這欄的形式，每期我們刊出問題的同時，亦將答案一併刊出。如此一來，可能令讀者失望，失去參加有獎遊戲的機會，但我們覺得這樣做法對於大部份讀者可能更有益處，因爲他們無需再等待下一期便能知道答案結果，方便研究程式中的蟲。當然，答案可能不是最完善的，我們歡迎讀者來信指正及改良。假如來信對讀者有一定的價值，我們將會在讀者園地一欄內將來信刊出。

雖然我們取消了這個有奬遊戲，但我們仍然鼓勵讀者自己設計一些「有蟲」的程式寄給我們，捉蟲訓練是公開給讀者的。程式長度沒有限制，當然越精簡越好，程式必需是用 APPLESOFT 語言，並需符合下列各項要求：

1、程式必須是 APPLESOFT ，任何 POKE 機械語言的子程式或 CALL 該子程式必須正確。
2、程式必須能出現有某些結果，而不會出岔子，換言之絕不能有 SYN—TAX ERROR出現。
3、程式必須附加簡短說明。
4、必須連同答案寄來。

假如閣下投寄來的捉蟲訓練程式是有水準，我們將贈閱十二期雜誌給你，或者再多一年的免費讀訂。

投稿請在信封面寫上「投稿捉蟲」。

至於第八期捉蟲問題的得獎者，將於第十期刊出，請各位留意。

今期的捉蟲問題是來自陳振權君。他的程式是印出1－200 內有質數。□

```
10  HOME : DIM N(200)
20  FOR A = 1 TO 200:N(A) = A: NEXT
     A
30  PRINT N(1);" ";:K = 0
40  FOR B = 2 TO 200
50  IF N(B) < 0 THEN 110:K = K +
     1
60  IF K > 5 THEN  PRINT
70  FOR C = B TO 200 STEP B
80 N(C) =  - 1
90  PRINT B;" ";
100  NEXT C
110  NEXT B
120  PRINT "THERE ARE ";K;" PRIME
      NOS."
```

此程式是希望將質數計出從每5個一横行，印畢後，在另一新行印出共有質數幾多個

```
如:  2 3 5 7 11
     13 17 19 23 29
     31 37 41 43 47
     53 59 61 67 71
     73 79 83 89 97
     101 103 107 109 113
     127 131 137 139 149
     151 157 163 167 173
     179 181 191 193 197
     199
     THERE ARE 46 PRIME NOS.
```

答案請看第86 頁。

第八期捉虫問題——匿藏起來的虫答案

匿藏起來的蟲程式中要求用家輸入亂數種子。APPLESOFT 中的種子必須爲負數，而取出亂數時的引數必須爲正數，任意正數的效果是相同的。至於零，則只是取回上次的亂數。故程式中若用家輸入負數或零，則只會產生一列相同的數，若輸入正數，則不會重新起動亂數，故有違輸入種子的目的。

此外，程式中首行以N＝10，用意必為方便修改程式，故日後修改後N可能大於10，應加上 DIM A (N) 這一句。

又程式中的浮泡法爲簡易的，讀者可用修改後的代之。

```
0   REM  ****************************
1   REM  *                          *
2   REM  * NOW THE BUG IS GONE !    *
3   REM  *                          *
4   REM  ****************************
5   REM
10 N = 10
20  DIM A(N)
30  TEXT : HOME : VTAB 10: INPUT
     "PLEASE ENTER THE SEED FOR T
     HE RANDOM FUNCTION : ";S
35  IF S = 0 THEN  PRINT  CHR$ (7
     ): GOTO 30
39 S =  ABS (S):Z =  RND ( - S)
40  FOR I = 1 TO N
50 A(I) =  RND (1)
60  NEXT I
63  HOME
65  VTAB 4: PRINT "UNSORTED ARRAY
     :": FOR I = 1 TO N: PRINT A
     (I): NEXT
70  FOR I = 1 TO N - 1
80  FOR J = N - 1 TO I STEP  - 1
90  IF A(J) > A(J + 1) THEN Z = A
     (J):A(J) = A(J + 1):A(J + 1)
      = Z
100  NEXT J: NEXT I
105  POKE 32,20: VTAB 4: PRINT "S
      ORTED ARRAY :": FOR I = 1 TO
      N: PRINT A(I): NEXT
110  POKE 32,0
```

□

任意設定索引顯示形式

本程式功能可以任意改變 CATA－LOG 顯示的模式。

當RUN 程式後，螢幕會出現一個選擇，總共有六項：

（1）改變 DISK VOLUME 的標題。

（2）改變 LOCK 檔案的符號。

（3）改變檔案種類的符號。

（4）列出或不列出佔用 SECTOR 的數值。

（5）列出或不列出檔案的名字。

（6）離開程式。

假如要中途改變選擇，可按 CTR－L－Q即可。

上述的改變，只是暫時貯放在RAM 內，當 REBOOT DOS後，改變便會消失，假如要將改變保持不變及留在磁碟內，可以將改變後的 CATA－LOG ，連同一個 HELLO 程式，再INIT一隻空碟，則有關的改變將永遠存在磁碟之上。

本程式在程式磁碟上的 FILENA－ME 是 READER'S CATALOG 。□

```
10  REM  ********************
20  REM  *                  *
30  REM  * READER'S CATALOG *
40  REM  *  BY COMPUHOUSE   *
50  REM  *                  *
60  REM  ********************
70  TEXT : HOME : PRINT "*
     ********* READER'S CA
     TALOG **********"
80  POKE 34,4
90  VTAB 6: INVERSE : HTAB 14: PRINT
     "CATALOG MENU"
100  NORMAL
110  PRINT : PRINT "1] CHANGE HEA
     DING";
120  PRINT  TAB( 20);"2] CHANGE L
     OCK SYMBOL"
130  PRINT "3] CHANGE TYPE";
140  PRINT  TAB( 20);"4] CHANGE S
     IZES"
150  PRINT
160  PRINT "5] CHANGE NAMES";
170  PRINT  TAB( 20);"6] QUIT"
180 CH =  PEEK ( - 16384)
190  IF CH > 175 AND CH < 183 THEN
     210
200  GOTO 180
210  POKE  - 16368,0
220  IF CH = 176 THEN  TEXT : HOME
     : END
230  IF CH = 177 THEN 290
240  IF CH = 178 THEN 410
250  IF CH = 179 THEN 500
260  IF CH = 180 THEN 640
270  IF CH = 181 THEN 720
280  IF CH = 182 THEN  END
290  HOME :: PRINT
300  PRINT "WHEN ASKED,PLEASE INP
     UT THE HEADING"
310  PRINT : PRINT "YOU WISH TO B
     E SHOWN WHEN THE DISK "
320  PRINT : PRINT "IS CATALOGED.
     PLEASE DO NOT INPUT"
330  PRINT : PRINT "MORE THAN 12
     CHARACTERS,OR IT MAY"
340  PRINT : PRINT "NOT WORK PROP
     ERLY ."
350  PRINT
360  INPUT "HEADING:";HD$
370  IF HD$ =  CHR$ (17) THEN 790
380  IF  LEN (HD$) < 12 THEN HD$ =
     HD$ + " ": GOTO 380
390  FOR X = 1 TO 12:H1$ =  MID$
     (HD$,X,1): POKE 46011 - X,( ASC
     (H1$) + 128): NEXT X
400  GOTO 790
410  HOME : PRINT
420  PRINT "WHEN ASKED,INPUT THE
     SYMBOL YOU WANT"
430  PRINT : PRINT "TO TAKE THE P
     LACE OF THE ASTERISK": PRINT
440  PRINT "FOR DENOTING A LOCKED
     FILE IN THE ": PRINT
450  PRINT "CATALOG OF THE DISK."
     : PRINT
460  PRINT : PRINT "SYMBOL'S LOCK
     FILES: ";: GET SY$: IF SY$ =
     CHR$ (17) THEN 790
470 SY =  ASC (SY$) + 128
480  POKE 44515,SY
490  GOTO 790
500  HOME : PRINT
510 : PRINT "WHEN ASKED, INPUT TH
     E SYMBOL YOU WANT": PRINT
520  PRINT "TO STAND FOR THE FILE
      TYPE MENTIONED.": PRINT
530  PRINT "SIMPLY PRESS ";: INVERSE
     : PRINT "RETURN";: NORMAL : PRINT
     " TO PASS.": PRINT
540  PRINT : POKE 34,14
550  HOME : PRINT "APPLESOFT FILE
     : ";: GET AF$:AF =  ASC (AF$
     )
560  HOME : PRINT "INVERSE FILE:
     ";: GET IN$:IN =  ASC (IN$)
570  HOME : PRINT "BINARY FILE: "
     ;: GET BF$:BF =  ASC (BF$)
580  HOME : PRINT "TEXT FILE: ";:
     GET TF$:TF =  ASC (TF$)
```

任意設定索引顯示形式

```
590  IF AF < > 13 THEN POKE 459
     93,AF + 128
600  IF IN < > 13 THEN POKE 459
     92,IN + 128
610  IF BF < > 13 THEN POKE 459
     94,BF + 128
620  IF TF < > 13 THEN POKE 459
     91,TF + 128
630  GOTO 790
640  HOME : PRINT
650  PRINT "YOU CAN EITHER:": PRINT

660  PRINT " 1)MAKE SIZES 000"
670  PRINT " 2)HAVE NO SIZES PRIN
     TED "
680  PRINT : PRINT "  CHOICE: ";
     : GET CH$: PRINT CH$:CH = VAL
     (CH$)
690  IF CH = 1 THEN POKE 44615,1
     69: POKE 44616,0
700  IF CH = 2 THEN FOR X = 4464
     3 TO 44645: POKE X,234: NEXT
     X
710  GOTO 790
720  HOME : PRINT
730  PRINT "YOU CAN EITHER:": PRINT

740  PRINT " 1)HAVE NO NAME PRINT
     ED"
750  PRINT " 2)HAVE NAMES SCRAMBL
     ED"
760  PRINT : PRINT "CHOICE:";: GET
     CH$: PRINT CH$:CH = VAL (CH
     $)
770  IF CH = 1 THEN FOR X = 4457
     1 TO 44573: POKE X,234: NEXT
     X
780  IF CH = 2 THEN POKE 44542,3
     2: POKE 44543,72: POKE 44544
     ,249
790  TEXT : HOME : VTAB 12: PRINT
     "ANY FURTHER CATALOG REFINEM
     ENTS?";
800  GET B$: PRINT B$: IF B$ = "Y
     " THEN 70
810  HOME : PRINT CHR$ (4);"CATA
     LOG": END
```

□

印出高解像數目字

這是一個非常之短但實用的子程式，可用來隨意在高解像頁裡編印零至10個數目字。使用時只需將要印的數字轉爲"N"，然後再 GOSUB 這條短程式即可。希望各位覺得有用。

本程式在程式磁碟的 FILENAME爲 HI—RES NO. □

```
HI-RES NUMBERS

1000  GOSUB 10000
1005  HGR : FOR X = 10 TO 260 STEP
      45: FOR Y = 5 TO 155 STEP 10
      :N = Y * X: GOSUB 9900: NEXT
      Y,X: END
9900  HCOLOR= 3: SCALE= 1: ROT= 1
      :N$ = STR$ (N): FOR J = 1 TO
      LEN (N$)
9910  DRAW VAL ( MID$ (N$,J,1)) +
      1 AT (X - 7) + (J * 7),Y: NEXT
      J: RETURN
10000 HIMEM: 37888
10005 POKE 232,0: POKE 233,148: FOR
      I = 37888 TO 38017: READ J: POKE
      I,J: NEXT : RETURN
10010 DATA 10,0,22,0,36,0,44,0,5
      4,0,66,0,76,0,88,0,99,0,108,
      0,119,0
10015 DATA 12,37,28,63,23,54,46
      ,30,14,45,5,36,4,0,36,188,15
      0,18,45,28,36,0
10020 DATA 101,228,63,23,150,24
      1,46,45,37,0,12,12,60,63,183
      ,146,21,45,12,228,7,0
10025 DATA 58,39,12,12,12,54,17
      4,55,62,0,56,39,44,45,245,17
      0,54,23,63,28,4,0
10030 DATA 117,246,63,28,36,229
      ,12,12,45,6,0,12,12,60,63,18
      3,82,30,46,00
10035 DATA 231,100,45,21,246,14
      ,246,63,28,36,0,231,100,45,2
      1,54,119,30,30,63,4,0
```

□

各期程式磁碟磁帶總目錄

1

1 突破傳統的鍵盤
2 捉球遊戲
3 勇闖電腦室
4 迷宮救美
5 將 APPLESOFT 變爲 FORTRAN
6 上、下、左、右移動文字畫面
7 測試工具— BOOLEAN 程式
8 騰空更多磁碟空間一搬移HELLO
9 重現刪除之檔案
10 電腦時代月曆
11 COMPUTEXT 文字處理
12 時日相距計算程式
13 增強英文字滙能力
14 幼童數學四則運算練習
15 過四關

2

1 貨櫃處理
2 沙蟹幻方局
3 巧破密令
4 美術字繪劃器
5 小階字母處理
6 程式轉載器
7 彩色電視／監視器 色彩測試
8 保留遊戲積分記錄
9 數學套件程式
10 磁碟修護工具
11 加送電腦時代月曆 圖像一個

3

1 DOS 的自動同步追踪器
2 動量不滅定律模擬法
3 自製磁碟索引表封套
4 蘋果編曲器
5 幼童認數字初階
6 賽跑遊戲
7 步步追踪
8 DOS KILLER
9 圍棋遊戲
10 巴哈的古典樂章名作
11 旋轉螢幕畫面
12 雜誌目錄分類歸檔系統
13 神經病醫生一神經鬆弛體操

4

(1) DOS 操作指令簡化器
(2) TEXT FILE 有些甚麼內容？
(3) 立體迷宮
(4) 英文串字練習
(5) 如何以機械語言來修補 DOS
(6) 電腦動畫製作原理。
(7) 豐富設計的學前兒童教育程式
(8) 應用數學套件程式(二)
(9) DOS WRITER
(10) 穿梭機哥倫比亞號機
(11) 找回失落了的 STEP 及 TRACE
(12) TRACER控制常式
(13) 將 RAM 卡作爲磁碟機使用
(14) 黑房計時器
(15) 太陽系模擬系統

5

1 私人約會備忘日曆
2 三度空間顯像系統
3 將 CP／M 檔案轉爲 DOS
4 蘋果萬花筒
5 高解像度文字產生器
6 修訂 DOS 操作指令簡化器
7 濃縮圖片儲存空間
8 幼童數學練習── 數字咭
9 加數機大登場
10 備忘 MEMO 紙
11 光速計數器
12 剩餘磁碟空 SECTOR 數目
13 16進制輸入數字鍵盤
14 問題棋盤
15 神偷智破夾萬
16 隔山打牛
17 大空船降落任務

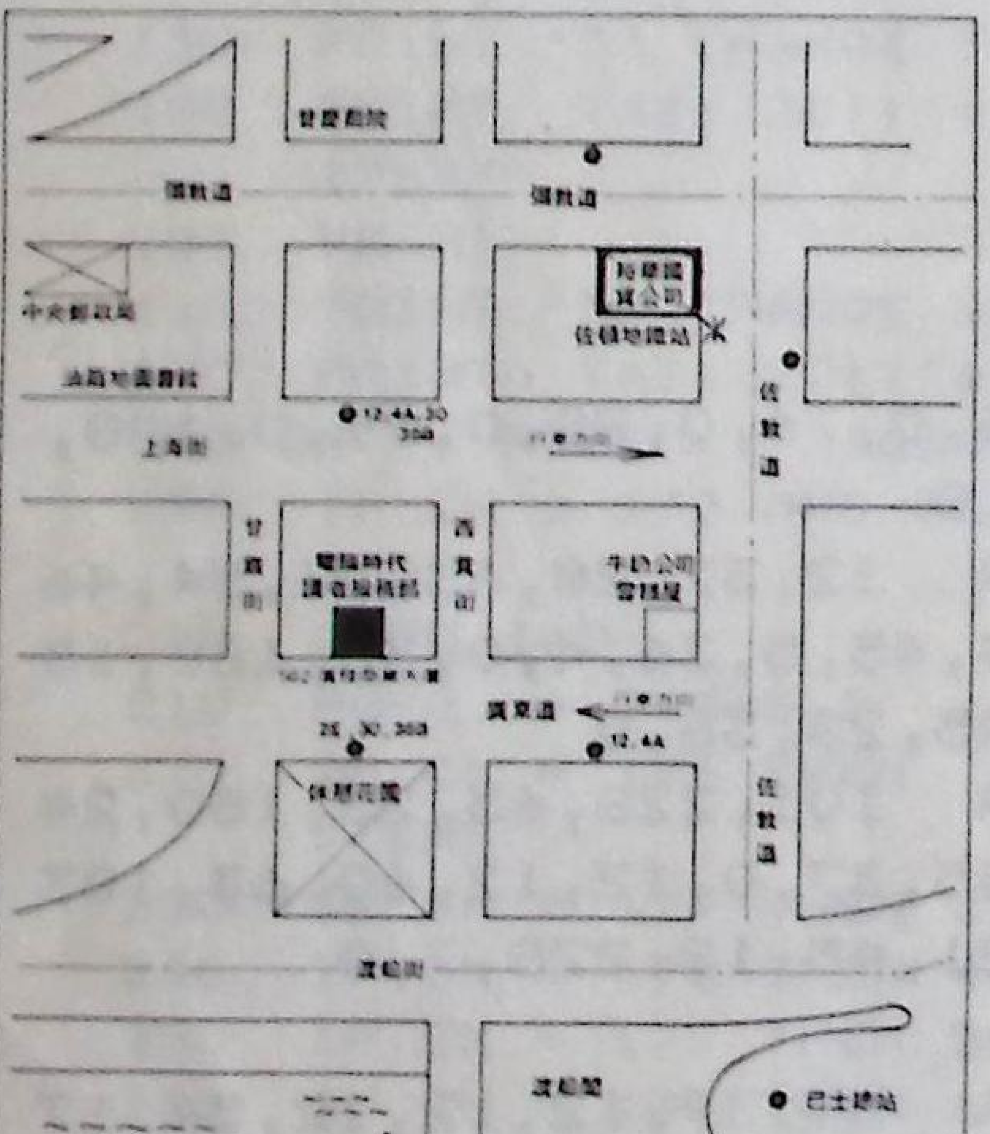
6

(1) 私人會計系統。
(2) APPLESOFT 擴展版本。
(3) GO ］［，一個取代 HELLO 的程式
(4) APPLESOFT 偵錯系統。
(5) DOS 指令再修訂版本。
(6) 布倫時鐘。
(7) AMPERSAND 數基轉換器。
(8) 完整索引錄。
(9) 砌積木。
(10) 神奇數字方塊。
(11) 亡命飛車。
(12) 迷宮搜秘。
(13) 太空怪侵略戰。
(14) 累加遊戲。
(15) 爲磁碟命名。
(16) 任意定義鍵盤。
(17) 捉蟲訓練問題。

精選程式磁碟

編號：CA00A
內容：遊戲及教育程式
遊戲程式
1、捉球遊戲（第一期）
2、勇闖電腦室（第一期）
3、迷宮救美（第一期）
4、過四關（第一期）
5、沙蟹幻方局（第二期）
6、貨櫃處理（第二期）
7、賽跑遊戲（第三期）
8、步步追踪（第三期）
9、圍棋遊戲（第三期）
10、立體迷宮（第四期）
11、穿梭機哥倫比亞號（第四期）
教育程式
12、增強英文匯能力（第一期）
13、幼童數學四則運算（第一期）
14、數學套件程式（第二期）
15、動量不滅定律模擬法（第三期）
16、幼童認數字初階（第三期）
17、豐富設計的學前兒童教育程式（第四期）
18、應用數學套件程式之二（第四期）

編號：CA00B
內容：應用程式精選
1、電腦時代月曆（第一期）
2、COMPUTEXT 文字處理（第一期）
3、美術學繪劃器（第二期）
4、彩色電視／監視器色彩測試（第二期）
5、保留遊戲積分紀錄（第三期）
6、自製磁碟索引封套（第三期）
7、APPLE 編曲器（第三期）
8、雜誌目錄分類歸檔系統（第三期）
9、DOS WRITER（第四期）
10、電腦動畫製作原理（第四期）
11、將 RAM 咭作爲磁碟機使用（第四期）



精選程式 磁碟及磁帶

讀者服務部由即日開始推出兩張全新製作的程式精選磁碟：一張是「遊戲及教育程式」，另一張是「應用程式」，兩者均是將電腦時代第一至第四期內精彩的程式融合爲一。每張訂價港幣 45 元。

與此同時，亦推出各期遊戲程式的磁帶版本，每盒訂價港幣十元。喜歡玩遊戲磁帶的讀者記著不要錯過。精選程式磁碟及第8期程式磁帶可在下列指定地點購買：

（1）百老滙冲印公司　第一店
九龍旺角彌敦道 758 A地下（近凱聲戲院）。電話：3—816931

（2）百老滙冲印公司　第十四店
香港銅鑼灣軒尼斯道 529號（三越百貨公司對面）電話：5—770181

或親臨電腦時代讀者服務部購買
查詢電話3—312007。

各期程式磁帶

編號：C#1
1、捉球遊戲
2、勇闖電腦室
3、迷宮救美
4、過四關
5、增強英文字滙能力
6、幼童數學四則運算
7、時日相距計算程式

編號：C#2
1、沙蟹幻方局
2、貨櫃處理
3、數學套件程式
4、巧破密令
5、彩色電視／監視器色彩測試

編號：C#3
1、賽跑遊戲
2、步步追踪
3、圍棋遊戲
4、動量不變定律模擬方法
5、幼童認數字初階

編號：C#4
1、豐富設計的學前兒童教育程式
2、應用數學套件程式之二
3、立體迷宮
4、穿梭機哥倫比亞號
5、英文串字練習

編號：C#5
1、加數機大登場
2、神偷智破夾萬
3、隔山打牛
4、太空船降落任務
5、蘋果萬花筒
6、幼童數學練習——數字咭

編號：C#6
1、累加遊戲
2、亡命飛車
3、迷宮搜秘
4、太空侵略戰
5、切積木
6、神奇數學方塊

歡迎補購各期程式磁碟磁帶或雜誌

請前往 電腦時代讀者服務部／COMPUHOUSE

佐敦廣東道582號廣發商業大厦八樓三室。TEL：3-312007

領先Apex助你掌握電腦！



私人及小組電腦証書課程

1 JUNIOR COMPUTER OPERATION (7—12歲)
2 COMPUTER OPERATION & BASIC PROGRAMMING
3 ADVANCED BASIC
4 DISK OPERATING SYSTEM (FILE HANDLING)
5 WORDSTAR AND CP/M OPERATING SYSTEM
6 DBASE II AND UTILITY PACKAGES
7 ASSEMBLY LANGUAGE，COBOL，FORTRAN OR PASCAL
8 PEACHTREE INVENTORY CONTROL SYSTEM
9 PEACHTREE ACCOUNT SYSTEM
10 PROGRAMMERS FULL CERT·TRAINING COURSE
11 電腦磁碟折銷，保護及原理
12 初級電腦裝嵌及修理

私人教授，天天開課。小組學習，興趣濃厚

領先私人電腦訓練中心　Apex Computer Training Centre

＊歡迎報名或電話查詢＊

地址：九龍深水埗青山道135號3樓11室（近九江街，金盟商場對面，電梯2字）
電話：3-7280889，3-7299187

PRODOS COURSE八月底開班

由於不少讀者對 PRODOS 極有興趣，故本公司特舉辦 PRODOS COURSE，與讀者交流 PRODOS 的新知識。

PRODOS COURSE 將於八月底開班，講授內容如下：

①將 DOS 3.3 上的 RANDOM ACCESS TEXT FILE 轉換成爲 PRODOS 格式下的 TEXT FILE。
②將 DOS 3.3 上的 APPLESOFT，BINARY，RE—LOCATABLE，TEXT FILE 以單磁碟機轉換爲 PRODOS 上的 FILE。
③ PRODOS 的使用方法，及其技術資料。
④ BASIC PROGRAMMING IN PRODOS。

本課程將提供講義，及工具程式磁碟。
課程共四小時，費用一百五十元正。
詳情請來電查詢：3—312007 □

道歉

上期磁碟KEYBOARD程式中，行號 1040 次一B字，現修訂如下：

```
1040  DATA KEYBOARD(W.T.TSANG),BR
      UNKEYBOARD(W.T.TSANG)
```

另（展覽會採用輕觸電腦入場登記）一文中，錯植主辦機構名稱爲香港教育促進中心，此乃電腦教育促進中心之誤。

〔VOCABULARY 1-2-3〕
教育軟件

電腦時代讀服務部設有〔VOCABULARY 1-2-3〕軟件的示範，讀者若想深入了解這套軟件，可到服務部參觀。

訂閱

（請用英文塡寫，方便電腦處理）

（只適用於本港訂戶）

信封面請註明「訂閱」

本人欲訂閱

☐電腦時代一年（十二期）HK$**96**

☐電腦時代＋CompuHouse磁碟一年（十二期）HK$**455**

姓名：______________

由第___期開始

地址：______________

電話：______________

附上劃線支票 No. ______________

支票收款人請塡寫 "Computing Age Publisher"

請寄九龍中央郵箱71193號

NO. ________

COMPUTING AGE

電腦時代

第九期
一九八四年八月十五日出版
本刊每月十五日出版
每本訂價港幣八元正

出　　版/ 電腦時代出版社
地　　址/ 廣東道582號廣發商業大廈703室
編　　輯/ 電腦時代編輯部
通訊地址/ 九龍中央郵箱71193號
通訊電話/ 3-312007
軟件支援/ CompuHouse
讀者服務部/ 廣東道582號廣發商業大廈八樓三室
承　　印/ 永恒印刷公司
葵涌工業街美安工業大廈12字樓A座
發　　行/ 同德書局
九龍西洋菜街193號

今期內容

□ Program saved on disk